ZIQIANG PU

EXPLOITING GAN AS AN OVERSAMPLING METHOD FOR IMBALANCED DATA AUGMENTATION WITH APPLICATION TO THE FAULT DIAGNOSIS OF AN INDUSTRIAL ROBOT

UNIVERSIDADE DO ALGARVE
Faculty of Sciences and Technology
2022

EXPLOITING GAN AS AN OVERSAMPLING METHOD FOR IMBALANCED DATA AUGMENTATION WITH APPLICATION TO THE FAULT DIAGNOSIS OF AN INDUSTRIAL ROBOT

Declaração de autoria de trabalho

Declaro ser o autor deste trabalho, que é original e inédito. Autores e trabalhos consultados estão devidamente citados no texto e constam da listagem de referências incluída.

I hereby declare to be the author of this work, which is original and unpublished. Authors and works consulted are properly cited in the text and included in the reference list.

(Ziqiang Pu)

Resumo

O diagnóstico inteligente de falhas baseado em aprendizagem máquina geralmente requer um conjunto de dados balanceados para produzir um desempenho aceitável. No entanto, a obtenção de dados quando o equipamento industrial funciona com falhas é uma tarefa desafiante, resultando frequentemente num desequilíbrio entre dados obtidos em condições nominais e com falhas. As técnicas de aumento de dados são das abordagens mais promissoras para mitigar este problema.

Redes adversárias generativas (GAN) são um tipo de modelo generativo que consiste de um módulo gerador e de um discriminador. Por meio de aprendizagem adversária entre estes módulos, o gerador otimizado pode produzir padrões sintéticos que podem ser usados para amumento de dados.

Investigamos se as GAN podem ser usadas como uma ferramenta de sobre amostra--gem para compensar um conjunto de dados desequilibrado em uma tarefa de diagnóstico de falhas num manipulador robótico industrial. Realizaram-se uma série de experiências para validar a viabilidade desta abordagem. A abordagem é comparada com seis cenários, incluindo o método clássico de sobre amostragem SMOTE. Os resultados mostram que a GAN supera todos os cenários comparados.

Para mitigar dois problemas reconhecidos no treino das GAN, ou seja, instabilidade de treino e colapso de modo, é proposto o seguinte.

Propomos uma generalização da GAN de erro quadrado médio (MSE GAN) da Wasserstein GAN com penalidade de gradiente (WGAN-GP), referida como VGAN

(GAN baseado numa matriz V) para mitigar a instabilidade de treino. Além disso, propomos um novo critério para rastrear o modelo mais adequado durante o treino. Experiências com o MNIST e no conjunto de dados do manipulador robótico industrial mostram que o VGAN proposto supera outros modelos competitivos.

A rede adversária generativa com consistência de ciclo (CycleGAN) visa lidar com o colapso de modo, uma condição em que o gerador produz pouca ou nenhuma variabilidade. Investigamos a distância fatiada de Wasserstein (SWD) na CycleGAN. O SWD é avaliado tanto no CycleGAN incondicional quanto no CycleGAN condicional com e sem mecanismos de compressão e excitação. Mais uma vez, dois conjuntos de dados são avaliados, ou seja, o MNIST e o conjunto de dados do manipulador robótico industrial. Os resultados mostram que o SWD tem menor custo computacional e supera o CycleGAN convencional.

Palavras chave: Redes generativas adversárias, aumentação de dados, diagnóstico de falhas, matriz-V, distância de Wasserstein fatiada, manipulador robótico, dados desbalanceados

Abstract

Machine learning based intelligent fault diagnosis often requires a balanced data set for yielding an acceptable performance. However, obtaining faulty data from industrial equipment is challenging, often resulting in an imbalance between data acquired in normal conditions and data acquired in the presence of faults. Data augmentation techniques are among the most promising approaches to mitigate such issue.

Generative adversarial networks (GAN) are a type of generative model consisting of a generator module and a discriminator. Through adversarial learning between these modules, the optimised generator can produce synthetic patterns that can be used for data augmentation.

We investigate whether GAN can be used as an oversampling tool to compensate for an imbalanced data set in an industrial robot fault diagnosis task. A series of experiments are performed to validate the feasibility of this approach. The approach is compared with six scenarios, including the classical oversampling method (SMOTE). Results show that GAN outperforms all the compared scenarios.

To mitigate two recognised issues in GAN training, i.e., instability and mode collapse, the following is proposed.

We proposed a generalization of both mean sqaure error (MSE GAN) and Wasserstein GAN with gradient penalty (WGAN-GP), referred to as VGAN (the V-matrix based GAN) to mitigate training instability. Also, a novel criterion is proposed to keep track of the most suitable model during training. Experiments on both the MNIST

and the industrial robot data set show that the proposed VGAN outperforms other competitive models.

Cycle consistency generative adversarial network (CycleGAN) is aiming at dealing with mode collapse, a condition where the generator yields little to none variability. We investigate the sliced Wasserstein distance (SWD) for CycleGAN. SWD is evaluated in both the unconditional CycleGAN and the conditional CycleGAN with and without squeeze-and-excitation mechanisms. Again, two data sets are evaluated, i.e., the MNIST and the industrial robot data set. Results show that SWD has less computational cost and outperforms conventional CycleGAN.

Keywords: Generative adversarial networks, data augmentation, fault diagnosis, V-matrix, sliced Wasserstein distance, industrial robots, imbalanced data

To the memory of my maternal grandparents

Acknowledgements

This work would not have been possible without the unconditional support of my supervisor José Valente de Oliveira. In his enthusiasm, his technical soundness, and his great efforts to explain things clearly and simply I found inspiration to bring this work to a good end. I still remember the word he told me "Speed cannot grant you a P.hD, but the quality will". This word has inspired me a lot to focus on the research itself in my Ph.D. program.

I am heartily thankful to my co-supervisor, Chuan Li, whose encouragement, amity and support were generously distributed when most needed. The exchange of ideas and encouragement that he offered me were crucial at a time when the work was in need of gaining momentum.

I cannot finish without saying how grateful I am to my family for providing a loving environment. Thanks to my parents, my father Pu Zhaojun and my mother Xiang Fang whom have always educated me to do my best in all matters of life. My father's hard-working spirit gave me the motivation to move forward, and my mother's meticulous care made me feel the warmth of the family. Thanks for providing financial support for my study abroad, giving me the opportunity to see the world.

I would like to thank the Chinese Embassy in Portugal and the China Scholarship Council for the study abroad scholarship, number 2021-A591, which is a recognition of my research. Besides, thanks to my best friend Tang Miao for keeping in touch with me on the internet.

Contents

List of Tables

List of Figures

List of Abbreviations

AE	Auto-encoder.
ANN	Artificial neural networks.
Adam	Adaptive moment estimation.
AC-GAN	Auxiliary classifier GANs.
AAE	Adversarial auto-encoder.
Bi-GAN	Bi-directional GANs.
CNN	Convolutional neural network.
CycleGAN	Cycle consistency GANs.
cGAN	Conditional GANs.
DT	Decision tree.
DBN	Deep belief network.
DBM	Deep Boltzmann machine.
DCGAN	Deep convolutional GANs.
EMD	Earth-Mover distance.
FT	Fourier transform.
FID	Fréchet inception score.
GANs	Generative adversarial networks.
IS	Inception score.
JSD	Jensen-Shannon divergence.
KL	Kullback Leibler.

LSGAN	Least square GANs.
MSE	Mean square error.
MLP	Multilayer perception.
MS	Mode score.
MS-SSIM	Multi-scale structural similarity.
MMD	Maximum mean discrepancy.
MC	The model compatibility score.
NN	Neural network.
NLP	Natural language processing.
PHM	prognostics and health management.
RF	Random forest.
RV	Rotate Vector.
RBM	restricted Boltzmann machines.
RL	Reinforced learning.
SVM	Support vector machine.
SVR	Support vector regression.
SAE	Sparse auto-encoder.
SMOTE	Synthetic minority oversampling technique.
SWD	Sliced Wasserstein distance
SW-CycleGAN	CycleGAN using SWD.
SFT	Short-time Fourier transform.
SGD	Stochastic gradient descent
TL	Transfer learning.
VGAN	V-matrix based GANs.
VAE	Variational auto-encoder.
WD	Wasserstein distance.
WGAN	Wasserstein GANs.
WGAN-GP	Wasserstein GANs with gradient penalty.
WPT	Wavelet package transform.

1

Introduction

In which we state the problem and the motivation for this work, present the research objectives, the contributions and draw a road map for the reading of the dissertation.

1.1 Context and Motivation

The industrial robot is being more and more adopted to facilitate harsh operations such as blanking, injection molding, die casting, assembling and testing [4]. Those operations may lead to failures occurring at the robot. Without appropriate mainte-

nance, it may result in economic loss and serious damage. Hence, fault diagnosis of industrial robots is a crucial task for machinery maintenance. A common way to detect different faulty states is by using experts' experience [5] or processing some amplitude-frequency conversion techniques [6, 7]. Machine learning based fault diagnosis usually applies techniques such as support vector machine (SVM) [8], logistic regression [9], Bayes classification [10], decision tree [11] and neural network (NN) [12].

Deep learning [13], the training of many layers of NN, has become a promising strategy for fault diagnosis [14]. Chen *et al*. [15] stated that the different acquired vibration signals may lead to effective diagnostic results. Particularly, a multi-sensory data fusion technique is proposed for the fault diagnosis of rotating machinery bearings. In this method, a sparse auto-encoder (SAE) with a deep belief network (DBN) is combined, called SAE-DBN scheme. The results demonstrate that the proposal can effectively identify the machine running conditions. Gong *et al*. [16] designed another fault diagnosis method based on a modified convolutional neural network (CNN) framework. This framework uses one-dimensional convolutional computing to deal with raw time series, and a non-linear SVM is adopted as the final classifier. Experimental results show that the proposed approach had a faster diagnosis speed and a higher accuracy than the conventional CNN. Long *et al*. [17] addressed a hybrid algorithm, named evolving echo state network (ESN). This algorithm combine ESN with a swarm optimizer for the fault diagnosis of machineries. They analysed the effectiveness of the algorithm by comparing with other scenarios such as SAE, CNN and ESN. Results show that the proposal is promising for complicated fault diagnosis problems.

The deep learning models [18] include but are not limited to auto-encoder (AE), DBN [19], deep Boltzmann machines (DBM) [20], or CNN [21]. Remarkable results have been reported on the application of such models. However, several issues remain to be clarified. These include a (probabilistic or information-based) interpretation for the data processing [22] inside such models and the amount of data required for training a given deep model for a certain task. This is particularly relevant as, without enough data, these methods do not work. In mechanical systems such as industrial

robots, acquiring data under faulty conditions is rather difficult. This is because, an industrial robot has a relatively low probability of failure due to its robustness. On the other hand, when it fails it is rather difficult to keep it in operation long enough in this faulty state. Consequently, the fault data is scarce when compared to healthy data (imbalanced) and might not meet the requirements for training deep neural networks. The above approaches need a large amount of data [23, 24, 25] to predict good results. Insufficient data will cause bias in the data-driven models such as deep learning models. Therefore, it is necessary to monitor the health state of this equipment with enough data set. That is to say, the above data issues become even more important.

To solve this problem, data augmentation techniques have been introduced to increase faulty data sets. Zhang *et al*. [26] employed the synthetic minority oversampling technique (SMOTE) for the rotating machinery fault diagnosis. Han *et al*. [27] integrated improved SMOTE with SVM for diagnosing bearing faults. Yu *et al*. [28] raised a multi-stage semi-supervised learning approach with the data augmentation technique for the fault diagnosis of rolling bearings. Recently the generative adversarial network (GAN) [29, 30, 31, 32, 33] has been proposed and shown to be a promising learning strategy to address data augmentation. Zhou *et al*. [34] used a global optimized with AE for faulty data generation. Pu *et al*. [35] exploited GAN as an oversampling method for the fault diagnosis of an industrial robot. The objective is to eliminate the imbalance between healthy and faulty data.

In the GAN framework, two (deep) models are trained to perform a zero-sum game. One of the models, the generator, aims at reproducing synthetic samples while the other model, the discriminator, tries to detect whether the samples were synthetic or from a given data set. In this framework, the two models are trained simultaneously, each improving its own performance.

GAN is suffering from three main problems [36] which are mode collapse, training oscillation and vanishing gradient, which limits the performance of GAN and makes the training difficult. Mode collapse refers to a lack of diversity in the generator, training oscillation is the oscillating loss value during training and the vanishing gradient

occurs whenever the discriminator becomes rapidly too accurate. The thesis aims at addressing these problems and to bring the GAN framework as suitable as possible for data augmentation to satisfy the needs of industrial robot fault diagnosis.

1.2 Research Aims

The objective of our research can be enunciated as follows:

- To study the data needed for deep learning, in GAN, in the context of the fault diagnosis of an industrial robot.
- To investigate the critical learning problem and possible improvements in GAN.
- To study the loss function of GAN in what concerns training convergence and stability.

1.3 Main Research Contributions

Our main research contributions can be summarized as follows:

- We have applied, for the first time, GAN in fault diagnosis of a real six-of-freedom industrial robot.
- We investigate the application of GAN to generate synthetic examples representing fault states for mitigating the presence of an imbalanced data set in a fault diagnosis task of the industrial robot. More concretely, GAN generates a synthetic wavelet packet transform (WPT) based feature vector of a vibrational signal as acquired by an accelerometer. A comprehensive study taking into account six different scenarios for mitigating the imbalanced data, including classical under and oversampling (e.g., SMOTE) methods, as well as for assessing the effect of factors such as generator selection, the number of training examples in each class, data shuffling in training data, the distribution used for sampling input random data and initial conditions.

Published paper: Pu Z, Cabrera D, Sánchez R V, Cerrada M, Li C and Valente de Oliveira J, "Exploiting Generative Adversarial Networks as an Oversampling Method for Fault Diagnosis of an Industrial Robotic Manipulator," Applied Sciences, vol. 10, pp. 7712, 2020. [37]

- For mitigating training oscillations, motivated by both the theoretical background and the empirical evidence obtained in classification and regression problems, a V-matrix based regularization is used within the conditional GAN (cGAN) framework. The V-matrix based criterion proposed by Vapnik *et al* [38, 39] generalizes the well-known and widely used mean square error (MSE) criterion. In the same vein, our proposed GAN framework, VGAN, generalizes both the MSE GAN [40] and the WGAN-GP frameworks. Also, a novel stop criterion like strategy that keeps track during training of the most suitable model is proposed (Section 4.2.3). The application of the proposed VGAN to an industrial robot fault diagnosis where the VGAN is used as a data augmentation tool to cope with an imbalanced data set. Results show that VGAN outperforms nine other scenarios including vanilla GAN, conventional regularization and SMOTE. Furthermore, the stop criterion like mechanism allows to obtain a monotonic increasing performance of the model during training and, when combined with the proposed regularization, yields the highest fault classification accuracy among all other scenarios.

  Published paper: Pu Z, Cabrera D, Li C and Valente de Oliveira J, "VGAN: Generalizing MSE GAN and WGAN-GP for robot fault diagnosis," IEEE Intelligent System, vol. 37, no. 3, pp. 65-75, 2022. [41]

- The sliced Wasserstein distance is applied, for the first time, in the development of unconditional and conditional CycleGANs aiming at smoother, faster, more efficient convergence while addressing mode collapse. To the best of our knowledge, it is the first time that either unconditional or conditional CycleGAN (either with Wasserstein or sliced Wasserstein distance) is used to transfer healthy states

to different fault states for addressing the imbalanced data problem in fault diagnosis of an industrial robot. A comprehensive set of experiments show that, for both the unconditional and the conditional cases, sliced Wasserstein distance outperforms classic Wasserstein distance in CycleGANs. For the unconditional case of the robot faulty data augmentation, the improvement in convergence efficiency can be greater than 2 (two) orders of magnitude.

Published paper: Ziqiang Pu, Diego Cabrera, Chuan Li, José Valente de Oliveira, "Sliced Wasserstein cycle consistency generative adversarial networks for fault data augmentation of an industrial robot," Expert Systems with Applications, vol. 222, 119754, ISSN 0957-4174, 2023. [42]

1.4 Thesis Outline

This dissertations is organized as follows:

- Chapter 2 states the problem of fault diagnosis with industrial robots. This chapter first gives a brief introduction to the significance of fault diagnosis. Then illustrates intelligent fault diagnosis, which is mainly based on artificial neural networks. Next, an introduction about the industrial robots is covered, including their transmission, perception systems, and the characteristic of the track of the dynamic movement. Then, the chapter notes the main challenge for the fault diagnosis of industrial robots. Finally, an experimental test rig of the industrial robot is established to obtain data.

- Chapter 3 illustrates the state of the art in GAN. The chapter begins with an introduction to the deep generative model and the associate research topics. Then the theoretical background and derivation of GAN, including the adversarial idea, loss functions, optimizations, and challenges are introduced. After this, we show some remarkable variants of GAN. Next, the evaluation metrics of GAN are introduced. In addition, the application of GAN and their open research are interpreted. Then, a case study of using a GAN as an oversampling method for an

imbalanced data set is presented. Finally, we statistically analyse the obtained results to assess the feasibility of the proposal.

- Chapter 4 proposes a V-matrix based GAN to mitigate training oscillation. The chapter first gives us an introduction and motivation about the loss function in GANs. Then it presents the used methodology, including V-matrix theoretical derivation, the improved version of MSE estimation, and the novel early stopping for selecting the best generator for data generation. Finally, we present experimental results and draw some conclusions.

- Chapter 5 considers another alternative loss function in CycleGAN named sliced Wasserstein distance (SWD) for fast convergence and stable training. The chapter first introduces GAN loss function and motivation for this work. Then, the theoretical knowledge background about SWD, both unconditional and conditional CycleGAN, the squeezed-and-excitation mechanism and a new metric for generator selection are also detailed. Next, we present experimental and qualitative results on both unconditional and conditional CycleGAN (with and without squeeze-and-excitation mechanisms) with two data sets (both the public data set of MNIST and the in-house industrial robot data set) to see the generalization ability of the proposal. Finally, we draw some conclusions.

- Chapter 6 summarizes the conclusions of our research and outlines some promising directions for future works.

2

The problem: Fault diagnosis of an industrial robot

In which we introduce fault diagnosis, the industrial robot, the existing challenges in the fault diagnosis area and illustrate the experimental apparatus of an industrial robot.

2.1 Introduction

With the advancement of science and technology and the popularization of modernization, mechanical equipment and production systems [43] gradually play an increasingly important role in developing the national economy and society. There might be unpredictable conditions in these systems, various parts of the industrial equipment with long-term operation may cause unforeseen failures, leading to system malfunctions [44]. These failures will reduce work efficiency and stop production in the worst case, resulting in considerable property losses and even accidents. In response to these problems, researchers and engineers assess the system's health state by monitoring and predicting [45]. Early fault diagnosis [46] is generally performed by professional maintenance personnel. Firstly, engineers should observe the operating state of the equipment and then test the abnormal changes in its noise, trajectory, temperature, vibration, and other parameters. Finally, through comparison with the normal state, the corresponding diagnosis results are obtained through empirical analysis [47]. However, this kind of detection is time-consuming and requires high specialized maintenance personnel. Therefore, with the continuous development of intelligent manufacturing, timely and accurate intelligent fault diagnosis of equipment failures has become a top priority.

A type of data-driven intelligent fault diagnosis flow chart is shown in Fig. 2.1. This has three main steps: signal acquisition, feature extraction and recognition and prediction. In the first step, each type of time series signal will be measured by sensors, e.g., the accelerometer and then collected by the data acquisition system. In the second step, some feature extraction methods, e.g., amplitude-frequency conversion technique, will be performed on the collected signals to get the main information of the signals. In the final step, some machine learning based methods are conducted to realize the recognition and prediction. Glowacz *et al*. [48] proposed a feature extraction using thermal images BCAoID (Binarized Common Areas of Image Differences), which is used for the fault diagnosis of electric impact drills. Zhang *et al*. [49] presented the adaptive and concise empirical Wavelet transform for the fault diagnosis of

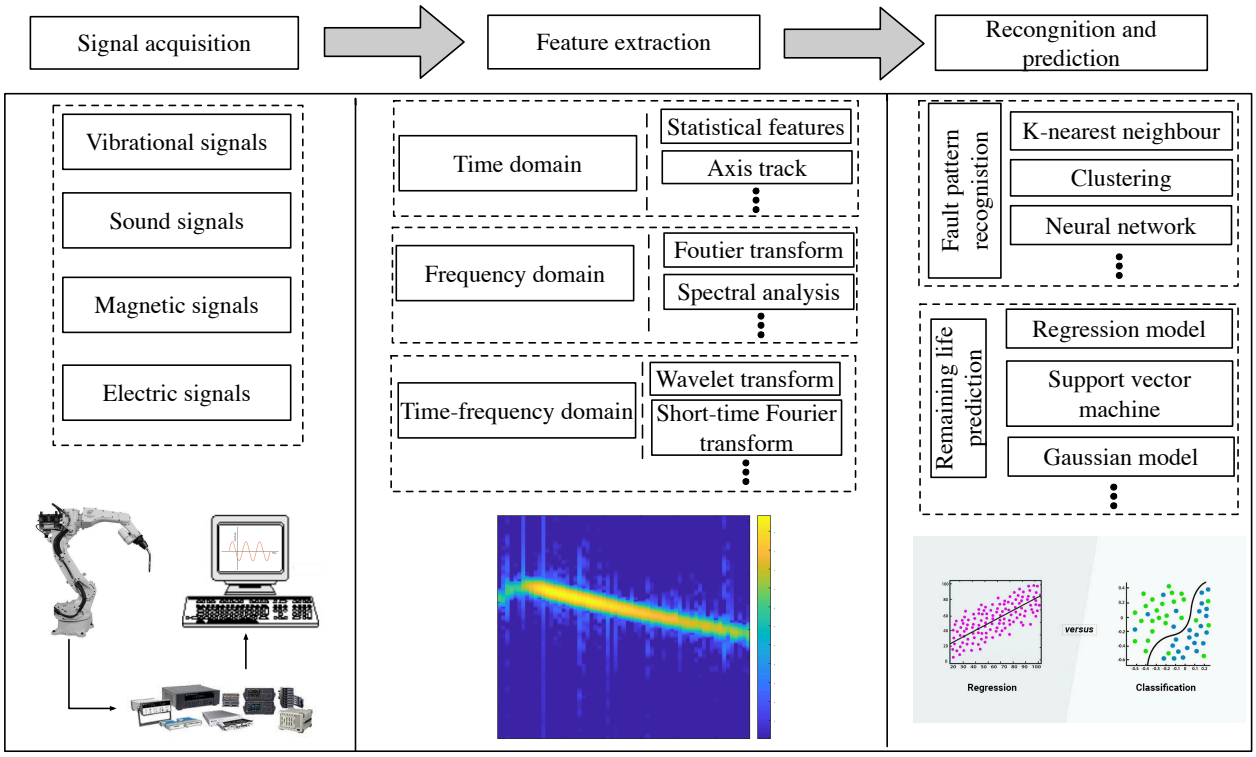


Figure 2.1: The flow chart of the fault diagnosis.

rolling bearings in rotating machinery. Due to the difficulty in realizing the fine diagnosis of motor faults under such high-speed, long-period, and heavy-load operations, Wu *et al*. [50] established a multilevel fine fault diagnosis method for fractional-order or integer-order faults on the basis of the powerful extraction ability of the fractional Fourier transform. Zhou *et al*. [51] proposed a detection method based on Transient-extracting transform and linear discriminant analysis for the fault diagnosis of rolling bearings. Nguyen *et al*. [52] used a NN with transformed vibration signals for bearing fault diagnosis. Han *et al*. [53] applied the idea of transfer learning (usually neural networks) to increase faulty data sets for machinery fault diagnosis. Pan *et al*. [54] designed a NN via restricted Boltzmann machines (RBM) with the layer-wise strategy for gearbox fault diagnosis. Tang *et al*. [55] applied CNN for the intelligent fault diagnosis of ratting machinery. Aydemir *et al*. argue that anomaly detection will improve the remaining useful life estimation of industrial machinery [56]. Lee *et al*. [57] proposed a method that combines CNN and a gated recurrent unit for anomaly detection of the rotating machinery. Dhiman *et al*. [58] developed anomaly detection based on an adaptive threshold and twin SVM. Besides, less research is reported about fuse in-

formation from the data of different sensors. To this end, Ma *et al*. [59] designed an information fusion method of the variational auto-encoder (VAE) and the random forest (RF) for fault diagnosis of rolling bearings. In this method, the signals measured by the accelerometer, the magnetic sensor, and the temperature sensor are fused for subsequent life evolution analysis. Applying the theory of artificial neural network to mechanical fault diagnosis for developing intelligent fault diagnosis is a new way of mechanical fault diagnosis. This intelligent fault diagnosis has been widely used and has become an important research direction in fault diagnosis.

Deep learning is now quite popular in fault diagnoiss. Wang *et al*. [60] proposed a novel deep CNN with multiple dimensions of signal features for bearing fault diagnosis. Firstly, the frequency domain signals are collected using the short-time Fourier transform and the wavelet transform. Then, the time domain signals, the frequency domain signals, and the time-frequency graph are fused into the model. Finally, the fault diagnosis task is performed to recognize the bearing position, damage location within the bearing, and the damage size. Du *et al*. [61] used the sparse isolation encoding forest to anomaly detection and novel detection on the wind turbine gearboxes.

Since the multiple fault variables and minor faults will bring obstacles to fault diagnosis, Zheng *et al*. [62] presented a fault detection technique for multivariate fault diagnosis. Firstly, the deviation factor area is adopted as the features of samples, and Bayesian decision theory is applied to calculate the probability of the equipment being faulty. Then, the multidimensional reconstruction-based contribution is used for fault identification. Besides, Pu *et al*. [63] developed a deep enhanced fusion network for the fault diagnosis for wind turbine gearboxes. The general idea is to fuse multi-channel signals with a feature extraction mapping for better feature capturing.

This chapter is organized as follows. In Section 2.2, we introduce some basic notions about industrial robots, including their transmission, perception systems and their characteristic of dynamic movements. In Section 2.3, we present the main challenge in intelligent fault diagnosis, especially in such a precision device as the industrial robot. In Section 2.4, we present the industrial robot considered in this thesis and describe the

performed data acquisition experiments. In Section 2.5, we draw some conclusions.

2.2 The industrial robot

Most industrial robots [64] are multi-joint manipulators or multi-degree-of-freedom robots. They can be seen as automatic actuators. In 1959, the world's first industrial robot was produced by Unimation [65]. With the progress of human society, industrial robots have been divided into three main generations: the first generation is the teaching-reproducing robot, the second generation is the sensory robot, and the last generation is the intelligent robot. With the recent changes in the global economic situation and the increase in labor costs, industrial robots are widely used in various industrial systems.

2.2.1 The transmission system

Regarding mechanical structure, industrial robots are generally divided into tandem robots (e.g., robotic arms) and parallel robots (e.g., 3-D printers) [66]. Tandem robots [67], such as six-degree-of-freedom industrial robots (with six independent movements), are characterized by the fact that the movement of one axis changes the origin of the other axis. In contrast, parallel robots [68] use a parallel mechanism in which the movement of one axis does not change the coordinate origin of the other axis. Tandem and Parallel robots are mainly connected by hinge connections. Unlike tandem robots, the Tandem robot is a closed-loop mechanism driven in parallel. The tandem and parallel robots are shown in Fig. 2.2.

The rotary joint of the tandem robot is the action point of the driving force of the robot movement, which is generally driven by the motor through the reducer. The reducer is a crucial part of the robot, and its cost accounts for about 1/3 of the cost of the robot body. Currently, two reducers are mainly used: the harmonic gear reducer and the rotate vector (RV) reducer. The harmonic drive technique was developed by C. Walt Musser *et al*. [69] in the mid-1950s. Teijin Corporation [70, 71] pioneered the

(a) (b)

Figure 2.2: The industrial robot: (a) Tandem robot; (b) Parallel robot [1]

development of RV reducers in the 1980s. The RV reducer consists of a front stage of a planetary gear reducer and a rear stage of a cycloid reducer. Compared with the harmonic gear reducer, the RV reducer has better rotation accuracy and accuracy retention. It is widely used in industrial robots and has become an indispensable core component of industrial robots with its stable transmission and high positioning accuracy.

2.2.2 The perception system

The perception system of the robot converts various internal state information and environmental information to data that the robot itself or between robots can understand and apply. The visual servo system uses visual information as a feedback signal to control and adjust the position and attitude of the robot. The application of this aspect is mainly reflected in the semiconductor and electronics industries. Machine vision systems are also widely used in quality inspection, work-piece identification, food sorting, and packaging.

2.2.3 The movement of the industrial robot

In order to let the robot complete some specific tasks in the shortest possible time and improve work efficiency, it is necessary to design a reasonable motion plan. Motion planning is divided into path planning and trajectory planning. Path planning aims to make the distance between the path and the obstacle as far as possible while the path length is as short as possible. The primary purpose of trajectory planning is to make the robot run as short as possible or the energy as small as possible in the movement of the robot joint space [72]. Trajectory planning adds time series information based on path planning. It plans the speed and acceleration of the robot when it performs tasks to meet the requirements of smoothness and speed controllability. Teaching reproduction is one of the methods to realize path planning. Teaching through the teaching box and recording the teaching results can intuitively and clearly show the running movement of the robot.

2.3 Challenge in the fault diagnosis of the industrial robot

The main challenge in the intelligent fault diagnosis is the difficulty in acquiring fault state data [73]. Due to the uncertainty of system faults, the problem of mechanical fault diagnosis becomes more complicated. Most of the current intelligent fault diagnosis research resort to labelled data [74]. However, labelling enough data becomes particularly difficult and time-consuming in the industrial field [75].

After acquiring data through sensors, the amount of health data is more significant than faulty data, resulting in an imbalanced data set. This can significantly cause deviations in intelligent fault diagnosis (usually based on artificial neural networks). A balanced dataset is needed for intelligent fault diagnosis. When the fault data is balanced by reducing the normal data, the health status of the equipment cannot be comprehensively analyzed [76]. Therefore, obtaining sufficient fault datasets plays a crucial role in fault diagnosis. To this end, many researchers use oversampling meth-

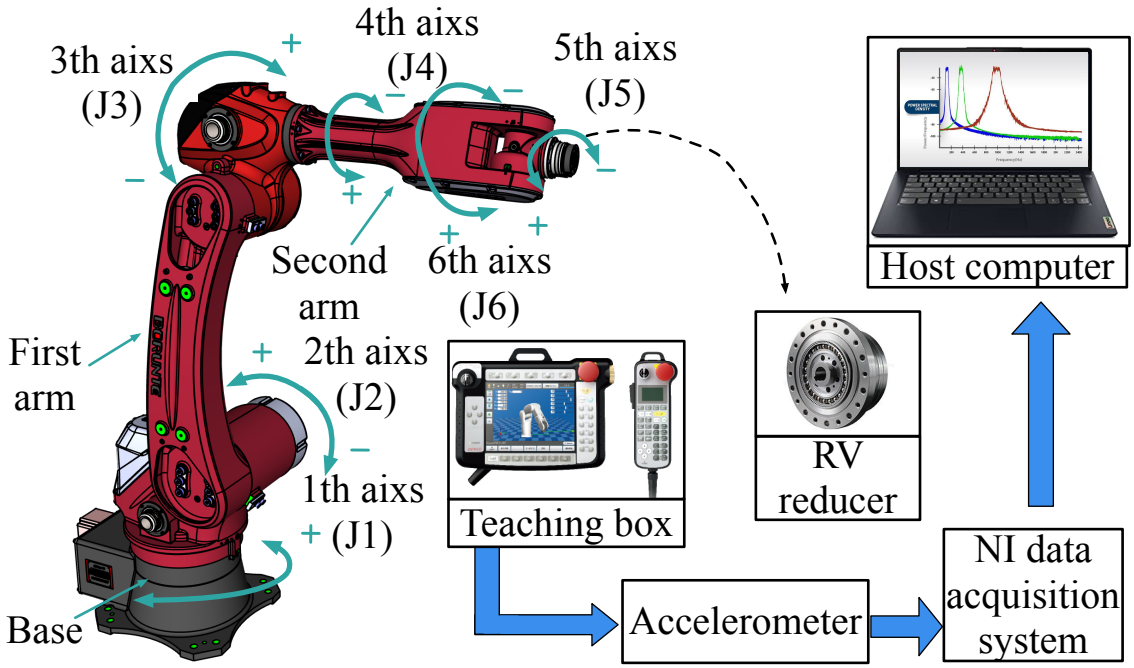


Figure 2.3: The experimental apparatus

ods to overcome this imbalanced problem in data acquisition by increasing the amount of fault data. Recently, GANs have been used for that reason

2.4 Experimental apparatus of the industrial robot

2.4.1 Experimental test rig

The main objective of fault diagnosis is to detect and classify incipient faults while the robot is operating. The test rig is shown in Fig. 2.3 and consists of an industrial robot (Brtirus 1510A), an accelerometer (PCB 622B01), a NI data acquisition system and a laptop (DELL XPS 9380). The accelerometer is mounted on the robot arm. The robot has six axes labeled from J1 to J6. Each axis has an RV reducer to operate the robot's dynamic movement. Therefore, the central part of dynamic monitoring for the fault diagnosis is on the RV reducer.

Table 2.1
Different fault patterns in the industrial robot.

Fault id	Part	Fault type
C_0	None	Healthy
C_1	Sun gear A	Pitting
C_2	Sun gear A	Broken tooth
C_3	Planetary gear B	Cracking
C_4	Planetary gear A	Cracking
C_5	Planetary gear B	Broken tooth
C_6	Sun gear B	Broken tooth
C_7	Sun gear A	Cracking
C_8	Planetary gear A	Broken tooth

2.4.2 Data measurement

During measurements, the working conditions of this industrial robot are with low-speed rotation of 600 r/min and a heavy load of 9.6 kg. The robot is moved by the motors, and the teaching box gives the instructions to the robot to start its next movement. At the beginning of the process, the robot is in its original position of 0 degree. Firstly, it will start back and forth movement from -115 degrees to 140 degrees of the limit range point in the first axis. Secondly, the same movement and the same limit range which is from -50 degrees to 35 degrees. Thirdly, the robot will move from -60 degrees to 90 degrees. Fourthly, the same configuration of movement is from -180 degrees to 180 degrees. Fifthly, the movement range will be decreased that the range is from -90 degrees to 90 degrees. At last, the robot will move from -180 degrees to 180 degrees and stop in the original place. This series of dynamic movements is only one experiment process. Next, we replace the faulty part to restart the above movement for the next experiment. Finally, the signal in each channel is collected by the NI acquisition system, which is an analog-to-digital conversion system that the digital samples are collected with an interface on the laptop. Note that all the data are collected in the School of mechanical engineering, Dongguan University of Technology, China.

Table 2.1 shows the each operating condition and Fig.2.4 shows an example of each one of the 8 types of faults. Measurements were performed at a sampling rate of 100 kHz. The sampling duration of each measurement is 20 s. The sampling interval was

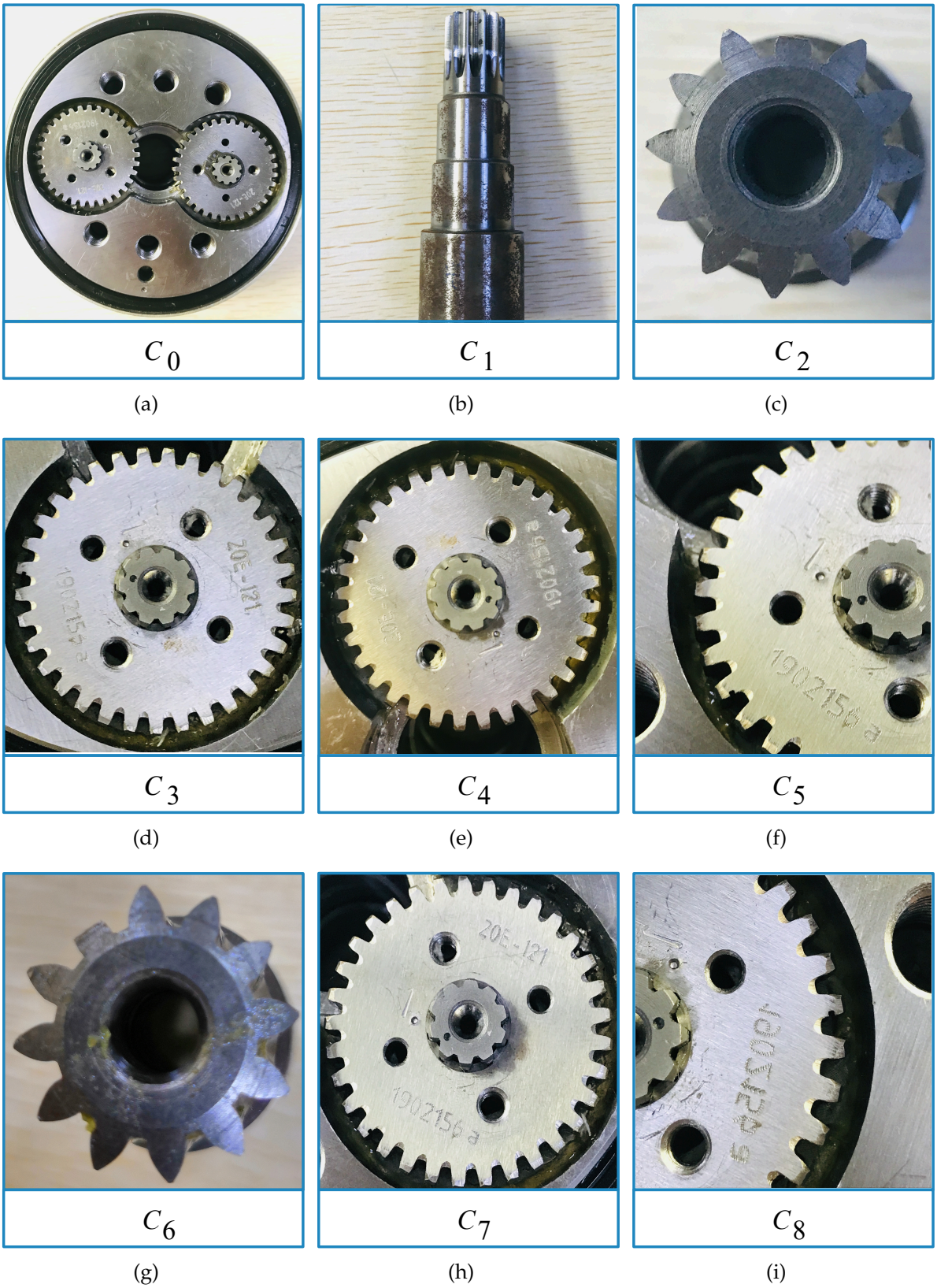
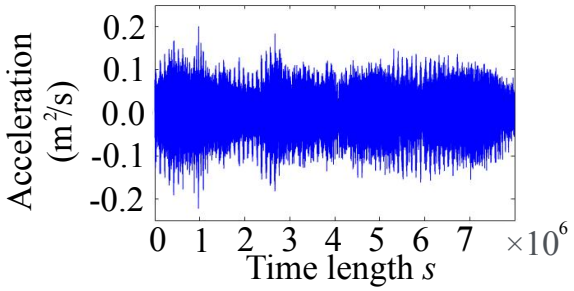
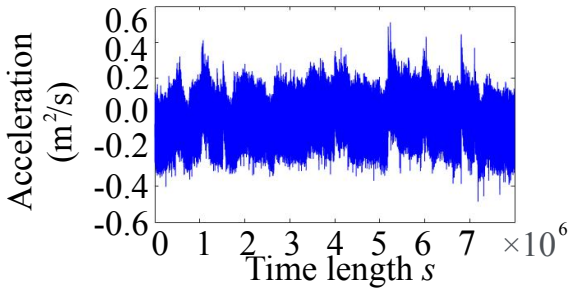


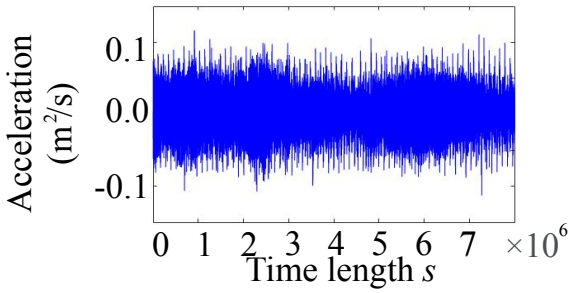
Figure 2.4: Examples of each one of the 8 monitoring conditions: (a) Healthy state; (b) Pitting in Sun gear A; (c) Broken tooth in Sun gear A; (d) Cracking in Planetary gear B. (e) Cracking in Planetary gear A; (f) Broken tooth in Planetary gear B; (g) Broken tooth in Sun gear B; and (h) Cracking in Sun gear A; (i) Broken tooth in Planetary gear A.


Acceleration (m²/s)
0.2
0.1
0.0
-0.1
-0.2
0 1 2 3 4 5 6 7 ×10⁶
Time length s

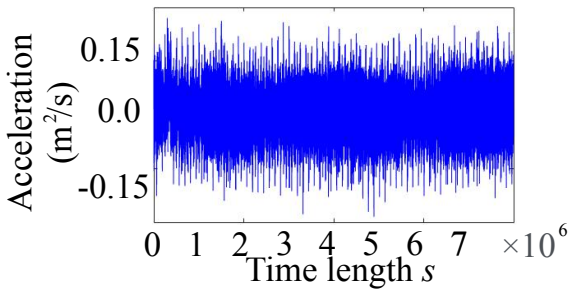
(a)


Acceleration (m²/s)
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
0 1 2 3 4 5 6 7 ×10⁶
Time length s

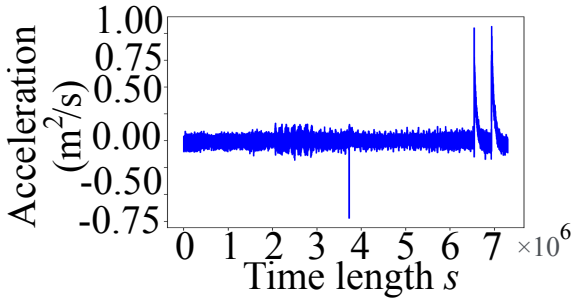
(b)


Acceleration (m²/s)
0.1
0.0
-0.1
0 1 2 3 4 5 6 7 ×10⁶
Time length s

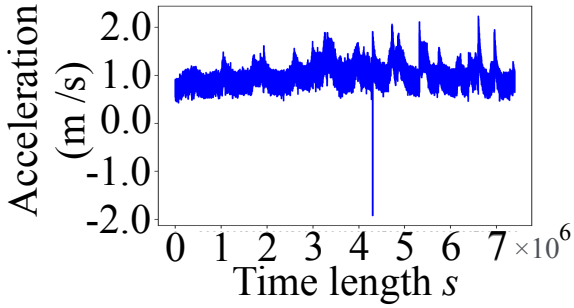
(c)


Acceleration (m²/s)
0.15
0.0
-0.15
0 1 2 3 4 5 6 7 ×10⁶
Time length s

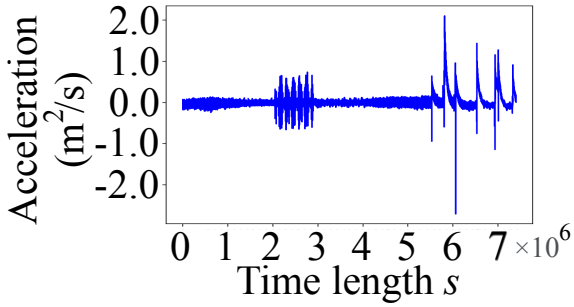
(d)


Acceleration (m²/s)
1.00
0.75
0.50
0.00
-0.50
-0.75
0 1 2 3 4 5 6 7 ×10⁶
Time length s

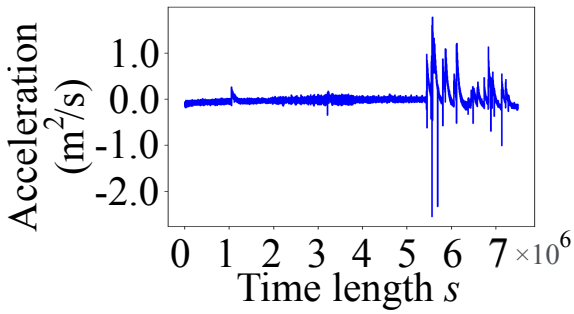
(e)


Acceleration (m /s)
2.0
1.0
0.0
-1.0
-2.0
0 1 2 3 4 5 6 7 ×10⁶
Time length s

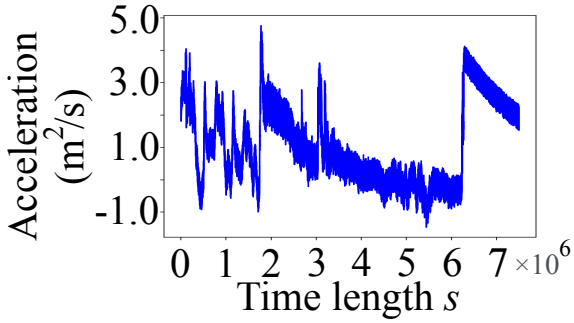
(f)


Acceleration (m²/s)
2.0
1.0
0.0
-1.0
-2.0
0 1 2 3 4 5 6 7 ×10⁶
Time length s

(g)


Acceleration (m²/s)
1.0
0.0
-1.0
-2.0
0 1 2 3 4 5 6 7 ×10⁶
Time length s

(h)


Acceleration (m²/s)
5.0
3.0
1.0
-1.0
0 1 2 3 4 5 6 7 ×10⁶
Time length s

(i)

Figure 2.5: The time length of vibration signal visualization in each fault condition: (a) Healthy condition (C_0); (b) Pitting in Sun gear A (C_1); (c) Broken tooth in Sun gear A (C_2); (d) Cracking in Planetary gear B (C_3); (e) Cracking in Planetary gear A (C_4); (f) Broken tooth in Planetary gear B (C_5); (g) Broken tooth in Sun gear B (C_6); (h) Cracking in Sun gear A (C_7); (i) Broken tooth in Planetary gear A (C_8).

Table 2.2
The number of examples available for each operating condition. C_0 stands for the class of nominal operating state, while C_1, C_2, and C_3 represent faulty states classes.

Class	C_0	C_1	C_2	C_3
Training set	14000	140	140	140
Validation set	6000	6000	6000	6000

set to 0.2 seconds. Thus, 20000 observations were obtained in each fault condition, and 20 k points were chosen for each observation. For hold-out validation, the data set was divided into two disjoint subsets, the training and the test (sub)sets. The training set has 70% of the data while the test set has the remaining 30%. Fig. 2.5 shows the time length of the vibration signal acquired in each one of the fault types.

2.4.3 Imbalanced data set

Since the fault diagnosis is critically dependent on the existence of a balanced, representative, large enough data set that is hard to obtain. With this in mind, we simulate such data scarcity in the experiments. That is, we start with less faulty state data than healthy data and use GAN as a data augmentation tool for obtaining the required balanced and representative data set. Four faults (C_0,C_1,C_2,C_3) were taken in to consideration which is detailed in Table 2.2 with an imbalanced data state.

2.5 Conclusion

In this chapter, we introduced the problem of the fault diagnosis of the industrial robot by elaborating on the machine learning based intelligent fault diagnosis, deep learning based intelligent fault diagnosis, the basic structure of the industrial robot, the challenges in fault diagnosis and the experimental apparatus of an industrial robot.

The core part of the industrial robot for dynamic movement is the RV reducer, especially the gearboxes. The fault gear will cause the wrong trajectory motion and bring troubles to the industrial process. Thus fault diagnosis becomes more necessary to find the faults in time. Traditional fault diagnosis is based on engineers' experience, which is time-consuming for maintenance. With the rise of artificial intelligence, the deep learning based intelligent fault diagnosis has been applied in the industrial field, usually based on ANN.

However, acquiring enough data in the industrial field is difficult. As a precision mechanical system like the industrial robot, its reducer (e.g., the RV reducer) will hardly work if damaged during operation. Without enough data, it will not be completely possible to monitor the health state of such devices. Thus, researchers need to find a way to increase the fault data set for the intelligent fault diagnosis of the industrial robot. With a balanced data set, the intelligent fault diagnosis can avoid deviation from the results, decreasing economic costs and saving time.

3

GAN Overview

In which we present a comprehensive survey of generative adversarial networks, elaborate on their role as a mean to data generation and address some key problems.

3.1 Introduction

The deep generative models such as DBN [77], RBM [78], denoising AE [79] and VAE [80] have shown their advantages for addressing significant effects on capturing the latent distribution of the data. Generally, generative modeling is an unsupervised learn-

ing task in machine learning that aims to discover hidden variables from data so that the model can be used to generate new examples. It has been widely used in images, speeches and videos. However, the above models generate blurry less than accurate samples.

In 2014, Goodfellow *et al*. [29] proposed generative adversarial networks (GAN) to capture data distributions for generative models. It is a powerful class of deep generative models to learn data distribution from a given random distribution, e.g., Gaussian distribution. GAN consists of two adversarial models (neural networks) [81]: a generator and a discriminator. The learning process can be described as a min-max game. The generator produces synthetic examples while the discriminator (like a binary classifier) tries to decide whether the current input is a real or a synthetic example. The goal of the generator is to deceive the discriminator by producing real-like samples that are indistinguishable from real ones. Both models improve their performance simultaneously up to a Nash equilibrium [82] using gradient-based optimization techniques [83, 84]. In the initial stage of training GAN, the discriminator can easily differentiate the generated data from the generator. With more iterations of adversarial learning between them, the discriminator cannot distinguish between two sets of samples, which means the generator is feasible to capture the underlying distribution of real samples.

The number of GAN applications has been steadily increasing. In the image processing field, Isola *et al*. [32] demonstrated that a conditional adversarial network is a promising approach for image-to-image translation tasks. Zhu *et al*. [33] presented an approach for learning to translate an image from a source domain to a target domain in the absence of paired examples. In the fault diagosis area, Li *et al*. [85] proposed a novel fault detection method for 3D printers using GAN, which consider only normal condition signals for training. Mao *et al*. [86] used GAN for an imbalanced data-driven fault diagnosis of rolling bearings. Pu *et al*. [87] developed a multi-functional framework with GAN for the anomaly detection of the industrial robot. Also, for rolling bearings, Jiang *et al*. [88] proposed a novel anomaly detection approach based on GAN with only health data. Li *et al*. [89] applied a GAN for the feature space learning in

fault diagnosis of 3D printers using only one sample in each faulty state. Wang *et al*. [90] proposed a method based on a conditional VAE and GAN for imbalanced fault diagnosis of the planetary gearbox. In the machine translation field, Zhang *et al*. [91] applied bidirectional GAN to tackle the exposure bias problem of machine translation. Experiment results on German-English and Chinese-English translation tasks demonstrate that this method can achieve significant improvements over baseline systems. Yang *et al*. [92] used GAN to build sentences for machine translation.

Despite its success in these applications, one problem with GAN is the training oscillation because of the adversarial learning between two neural networks. This critical challenge in the training phase often separates mere attempts from successful applications. In addition, if the discriminator can easily recognize the generated samples from the generator, it means that the discriminator's gradient has vanished and cannot provide any gradient for the generator to be updated. Besides, the mode collapse problem also happens in GAN, that is the trained generator only learns part of the distribution resulting in synthetic results with little or none diversity.

This chapter is organized as follows. In Section 3.2, we review the background of GAN. In Section 3.3, we introduce the theoretical basis of GANs variants and explain how GANs variants can address some issues in GAN. In Section 3.4, we describe some evaluation metrics. In Section 3.5, we present some application scenarios with GAN for image, machine translation, and industrial machinery fault diagnosis field. In Section 3.6, a case study is introduced about GAN as an oversampling tool for data augmentation. In Section 3.7, we summarize some relevant conclusions.

3.2 Theoretical Background on GANs

3.2.1 Artificial Neural networks

Artificial Neural Networks (ANN) [93] are complex structures inspired by the function of the brain. It is a simulation of the biological nervous system, and the information processing function is defined by the input and output characteristics (activation char-

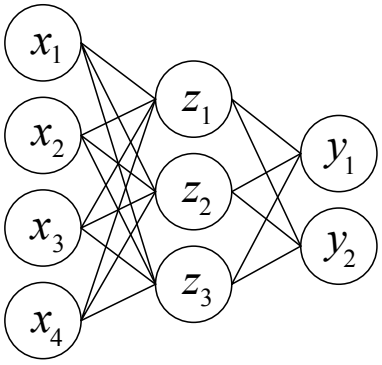


Figure 3.1: The architecture of multilayer perceptron with three layers: x stands for the input neurons, z is the hidden neurons and y represents the output neurons. See text for details.

acteristics) of the units of the network, the topology of the network (neuron connections), and it has strong linear characteristics [94]. These networks, including multilayer perceptron (MLP) and convolutional neural networks (CNN) can perform model function estimation and handle linear or non-linear functions by learning from data.

(a) Multilayer perceptron

MLP is a powerful modelling tool to generate a set of outputs from inputs with a set of the fully connected layer. Generally, MLP is a feedward ANN, which has at least three layers (an input layer, a hidden layer and an output layer) shown in Fig. 3.1. Each circle in Fig. 3.1 stands for a neuron. Given a set of input neurons x_i $(i = 1, 2, 3, 4)$, they will map to the hidden layer to obtain the hidden neurons z_1, z_2 and z_3. Finally, the hidden neurons will map to the output layer to get the output neurons y_1 and y_2. Each layer utilizes a transform, e.g., the $f(\cdot)$ in (3.1), to get the outputs from given inputs x_i and weights $\{w_i\}$. A backpropagation technique is used for MLP optimization.

$$y = f(\sum_{i=0}^{N} w_i x_i) \tag{3.1}$$

(b) Convolutional neural networks

CNN can perform better in training than MLP when more layers are increased. It is a kind of neural network specifically used for image recognition and tasks involving

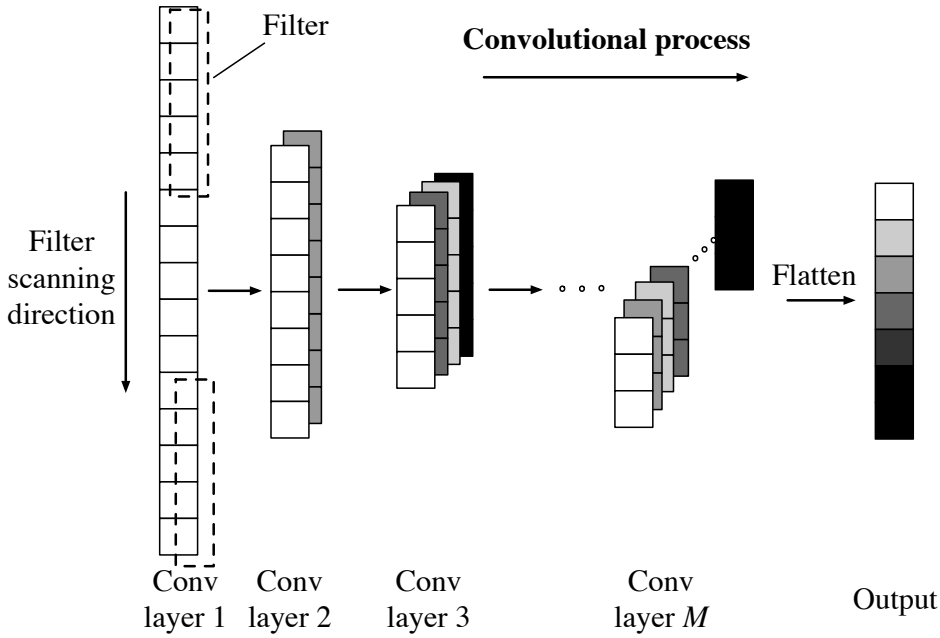


Figure 3.2: The schematic of convolutional process

pixel data processing. The basic convolutional process is shown in Fig. 3.2. At the first convolutional layer, the filter is moved from the top to the bottom. Secondly, the dot product is conducted in each shift. The shift for this matrix multiplication operation is called stride. For instance, with 1 stride, the filter will move 1 block. After the dot product operation, the value in each block will sum up to a total value with biases that are represented as the input for the next convolutional layer. Finally, after several convolutional processes, all the acquired vectors are flattened into one single vector which is the output of CNN. Given the raw data x=$\{$ x^1, x^2, x^3,..., x^m $\}$ $\in$ R^m. The convolutional layer involves all the inputs with convolutional kernels and then is followed by the activation $f(\cdot)$ to generate the output for the next convolutional layer. This can be formulated by

$$x^{i'}(l) = f(\sum_{i=1}^{m}(k^{ii'(l)} \cdot x^{i(l-1)})_{(k_s,s)} + b^{i'(l)}) \tag{3.2}$$

where k stands for the convolution kernel, $b^{i'(l)}$ is the l-th bias, l is the number of layers in the network, i=$\{$1,2,3,...,m $\}$ is the index of the input dimensions, $i^{'}$ = $\{$ 1,2,3,...,$m^{'}$ $\}$ is the index of the output dimensions, $\cdot$ denotes the dot product, k_s is the kernel size of k, and s is the stride option that the kernel will move every s steps on the input data.

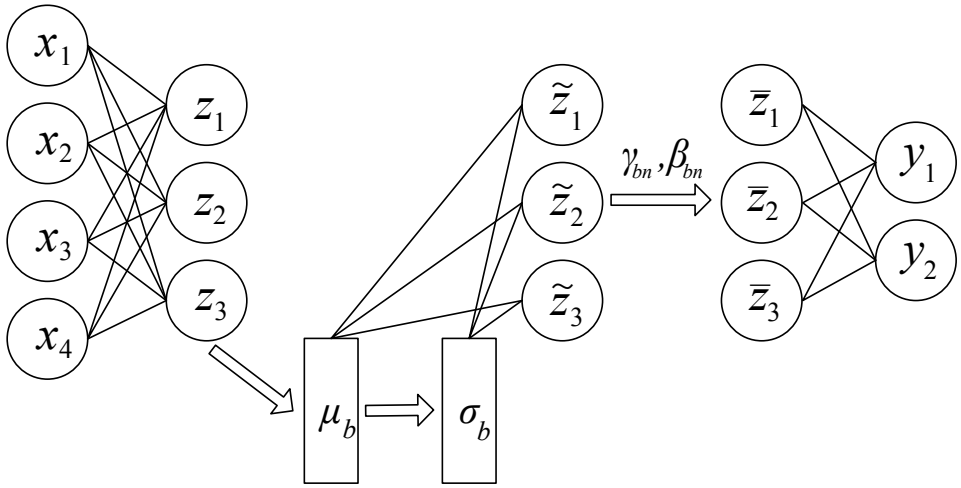


Figure 3.3: Neural network with batch normalization: x is the input neurons, z stands the hidden neurons, $\tilde{z}$ is the normalized hidden neurons, $\tilde{z}$ is the hidden neurons with batch normalization, y denotes the output neurons, μ_b and σ_b are the mean and the standard deviation of the hidden neurons z and $\bar{z}_i$ and β_{bn} are learnable parameters in the batch normalization. See text for details.

(c) Normalization techniques

During the backpropagation of ANN, the weights of each neuron are updated using the gradient of the loss function. With large number of layers for training in ANN, it will cause the vanishing gradient problem, which prevents the network from training. One reason for this difficulty is the input distributions may change after each mini-batch when the weights are updated. A mini-batch is a subset of the training dataset that reduce the variance of the gradient when using gradient descent based optimization. Batch normalization (BN) can help the convergence and avoid vanishing gradient when training a neural network. It was first proposed by Sergey *et al*. in 2015 [18]. The structure of the classical neural network with BN is shown in Fig. 3.3.

As shown in Fig. 3.3, the only difference is that the data in each layer will be normalized before propagation into the next layer. Given a set of neurons x_i $(i = 1, 2, 3, 4)$, they will map to the next layer to get hidden neurons z_i(z_1,z_2 and z_3). Then, BN will be performed on this hidden vector to get the mean and standard deviation of z_i known as μ_b and σ_b, respectively. Next, the newly hidden vector $\tilde{z}_i$ is calculated by the mean and standard deviation of z_i. Finally, the output of BN ($\bar{z}_i$) can be acquired with appropriate γ_{bn} and β_{bn}. Therefore, the idea of BN is to scale the data firstly through the

mean and variance and then scaling and shifting with parameters, e.g., γ_{bn} and β_{bn}. The whole process of batch normalization can be formulate as

$$\mu_b = \frac{1}{m}\sum_{i=1}^{m} z_i i = \{1,2,3\}, \tag{3.3}$$

where μ_b stands for the mean of z_i.

$$\sigma_b = \sqrt{\frac{1}{m}\sum_{1}^{m}(z_i - \mu_b)^2}, \tag{3.4}$$

where σ_b stands for the standard deviation of z_i.

$$\tilde{z_i} = \frac{z_i - \mu_b}{\sigma_b} \tag{3.5}$$

Even though the data has been scaled, the learned distribution in each layer will be changed. To this end, the following equation can avoid this problem

$$\bar{z_i} = \gamma_{bn}\tilde{z_i} + \beta_{bn} \tag{3.6}$$

where $\bar{z_i}$ is the final output of batch normalization, γ_{bn} and β_{bn} are the scaling and shifting parameters. These two parameters are learnable parameters during neural network training ensuring the accurate normalization of each mini-batch.

3.2.2 Adversarial idea

A GAN is based on the adversarial learning of two neural networks, a generator G and a discriminator D. Both G and D play a min-max game where the goal of G is to produce generated samples similar to real samples, and the goal of D is to discriminate the samples generated by G and samples from the real data distribution.

In order to learn the G's distribution p_g over data x, a prior random distribution (say Gaussian distribution) is defined as p_z and z is a set of noise vectors. Then, G maps from noise p_z to data space p_{data}. $G(z, \theta_g)$ is a neural network with weights θ_g.

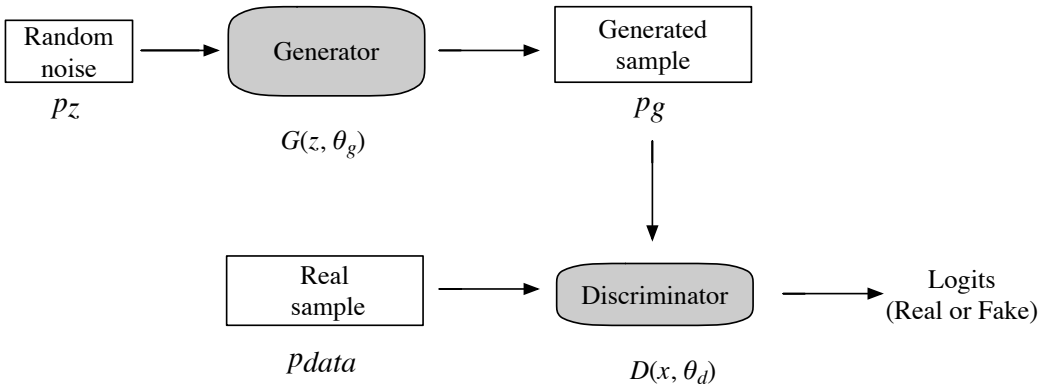


Figure 3.4: The architecture of a generative adversarial network(GAN).

The $D(x, \theta_d)$ is defined with weights θ_d and will define whether the output of $D(x)$ is from p_g or p_{data}. The discriminator D outputs the probability that x is from the data distribution rather than the generator G. The discriminator D is trained to maximize the probability of giving the correct label to both samples and generated samples. The generator G aims to minimize $log(1 - D(G(z)))$ (See Figure 3.4).

3.2.3 Loss function

The loss function of GAN is considered as

$$\min_{\theta_g} \max_{\theta_d} V(G, D) = \min_{G} \max_{D} E_{x \sim p_{data}}[logD(x, \theta_d)] + E_{z \sim p_z}[log(1 - D(G(z, \theta_g)))] \quad (3.7)$$

where $V(G, D)$ is a binary cross entropy function. The gradient-based optimization is used for updating θ_g and θ_d that is solved via the following two gradient updates.

$$\theta_g^{t+1} = \theta_g^t + \lambda^t \nabla_{\theta_g} V(D, G) \quad (3.8)$$

$$\theta_d^{t+1} = \theta_d^t + \lambda^t \nabla_{\theta_d} V(D, G) \quad (3.9)$$

where θ_g and θ_d are the weights of G and D, respectively. λ is the learning rate, and t denotes the iteration.

3.2.4 Optimization strategy

Gradient-based optimizations such as adaptive moment estimation (Adam), Root Mean Squared Propagation or stochastic gradient descent (SGD) are widely applied in neural networks to find the minimum of the performance index. The discriminator D and the generator G are trying to minimize their own loss function. For the discriminator D, the loss function is as follows

$$loss_D = \max_{\theta_d} E_{x\sim p_{data}}[logD(x)] + E_{z\sim p_z}[log(1 - D(G(z)))], \tag{3.10}$$

And the loss function of the generator G is given by

$$loss_G = \min_{\theta_g} E_{z\sim p_z}[-log(D(G(z)))], \tag{3.11}$$

The min-max game denotes the solution includes minimization and maximization, therefore,

$$loss_D + loss_G = 0 \tag{3.12}$$

As the loss function of the discriminator, $loss_D$ can be seen as

$$loss_D(\theta_d, \theta_g) = -\frac{1}{2}\int p_{data}(x)[logD(x)]dx - \frac{1}{2}\int p_g(x)[log(1 - D(x))]dx \tag{3.13}$$

The above equation (3.13) represents the classical cross-entropy optimization minimized during the training of a binary classifier with a *sigmoid* output. In the min-max game, the generator G attempts to fool the discriminator by minimizing and maximizing the log probability of the discriminator D. Suppose $y = 0$ means generated data and $y = 1$ denotes real one, the density ratio between the real sample and generated sample in GAN is represented as follows.

$$D(x) = \frac{p_{data}(x)}{p_g(x)} = \frac{p(x|y=1)}{p(x|y=0)} = \frac{D^*(x)}{1 - D^*(x)}, \tag{3.14}$$

Through training D, it learns to distinguish samples from data for any given G.

Combined with Eq. (3.13), the optimal D^* is searched by the derivative

$$\frac{dloss_D(\theta_d,\theta_g)}{dD(x)} = \frac{1}{2}\int p_{data}(x)\frac{1}{D(x)}dx - \frac{1}{2}\int p_g(x)\frac{1}{1-D(x)}dx, \tag{3.15}$$

From the necessary condition $\frac{dloss_D(\theta_d,\theta_g)}{dD(x)} = 0$, we have

$$\frac{1}{2}\int p_{data}(x)\frac{1}{D(x)} = \frac{1}{2}\int p_g(x)\frac{1}{1-D(x)}dx, \tag{3.16}$$

Or

$$p_{data}(x)\frac{1}{D(x)} = p_g(x)\frac{1}{1-D(x)}dx, \tag{3.17}$$

And it can be re-written as

$$D^* = \frac{p_{data}(x)}{p_{data}(x)+p_g(x)}. \tag{3.18}$$

After sufficient training iterations, G and D will converge to p_g=p_{data}. This denotes that the range of $D^*(x)$ is between 0 to $\frac{1}{2}$. Next, let $D^*(x)$=$\frac{1}{2}$ in (3.7) and we have

$$\max_{\theta_d} V(G,D) = -2log2 + 2(D_{JS}(p_{data}||p_g)), \tag{3.19}$$

where JS denotes the Jensen-Shannon (JS) divergence. Suppose there are two distributions p and q, the JS can be defined as

$$JS(p||q) = \frac{1}{2}KL(p||\frac{p+q}{2}) + \frac{1}{2}KL(q||\frac{p+q}{2}), \tag{3.20}$$

where KL is the KullbackLeibler (KL) divergence which can be shown as

$$KL(p||q) = \int p(x)log\frac{p(x)}{q(x)}dx. \tag{3.21}$$

However, finding the Nash equilibrium in GAN is very challenging as loss functions are non-convex, parameters are continuous, and the parameter space is high-dimensional [36], e.g., an update to θ_d that reduce $Loss_D$ can increase $loss_G$ and vice

versa.

3.2.5 Challenges in GAN

GAN suffers from limitations such as a lack of diversity, leading to generating similar or redundant samples [95], which is the so-called model collapse. Besides, the generator G and the discriminator D oscillate during adversarial training. When a player (say discriminator) gets more powerful than the other player (say generator), it may be possible that the system does not learn and can not provide gradients for updating relative weights (vanishing gradients) [96]. Also, the stop criterion for the generator selection is essential for GAN to get the best generator for optimal data generation. In this subsection, we will explain the above challenges in GAN.

(a) Oscillating convergence curves and gradient vanishment

Traditional deep learning-based models are trained alone to reach the lower loss value using gradient-based algorithms. In GAN, the loss value oscillates during training since there are two adversarial neural networks. In detail, the loss function of the generator G is $E_z[logD(G(z))]$ while the loss function of the discriminator D is $E_z[log(1 - D(G(z)))]$. The former loss function can cause gradient vanishing problems when D can easily differentiate between real and fake samples. The minimization of G's loss function equals minimizing KL divergence, which causes unstable gradients.

(b) Mode collapse

Mode collapse problems can occur in GAN when the generator loses diversity; a consequence of poor generalization. There can be two types of mode collapse: (1) most modes from the input data are absent from the generated data, and (2) only a subset of particular modes is learned by G. An ill-suited objective function can be a significant reason for the mode collapse problem where several GAN variants, including modifying D's objective [97, 98] and modifying G's objective [99], have been proposed. G is shown at equilibrium in these variants and can learn the whole data distribu-

tion, but convergence is elusive in practice. To handle this issue, several recent studies have introduced new network architectures with new objective functions or alternative training schemes.

(c) Stop criterion for GAN

GAN model has been used for wide applications in deep learning. However, its evaluation tools are still qualitative (i.e., visual examination of samples by a human) even though several approaches and measures have been introduced to evaluate GAN's performance. Visual inspection is time-consuming, subjective and cannot capture distributional characteristics, which is an important factor for unsupervised learning. As selecting an appropriate model is essential for getting good performance for an application, selecting relevant evaluation metrics is vital for drawing the correct conclusion. Designing a better GAN model requires overcoming the limitations of the qualitative stop criterion by developing or using proper quantitative metrics.

3.3 Important variants of GAN

There are many variants of GAN that have been created over last years. The major innovations include model architecture, training strategies and loss functions. In this section, we will introduce some important variants of GAN.

3.3.1 Deep convolutional GAN

Traditional GAN is defined using MLP. In deep convolutional GAN, the generator G and the discriminator D are deep CNN (DCGAN) [100], which have better performance on feature extraction. DCGAN has the same flow-chart of training as the conventional GAN.

There are three main features in DCGAN: (1) The overall architecture is mainly based on the all-convolutional net [101]. This architecture has neither pooling nor unpooling layers. When G needs to increase the spatial dimensionality of the represen-

tation, it uses transposed convolution (deconvolution) with a stride greater than 1; (2) Utilize batch normalization for most layers of both G and D. The last layer of G and the first layer of D are without batch normalization so that GAN can learn the correct means and scales of data distributions; (3) The adam optimization is utilized instead of SGD.

3.3.2 Least square GAN

The least square GAN (LSGAN) in [102] are designed to address the vanishing gradient problem in the original GAN. This work reported that the decision boundary for the discriminator D of the original GAN penalizes minimal errors to update the generator G for those generated samples far from the decision boundary. LSGAN uses the least squares loss instead of the cross-entropy loss (the JSD-based loss) in the original GAN. The $a-b$ coding is used for the LSGAN discriminator, where a and b are the labels for generated and real samples, respectively. There are two advantages of LSGAN compared to the conventional GAN: (1) The new decision boundary produced by D penalizes large errors to those generated samples far from the decision boundary, which makes those low quality generated samples move toward the decision boundary. This is good for generating higher-quality samples; (2) Penalizing the generated samples far from the decision boundary can supply more gradient when updating G, which overcomes the vanishing gradient problems in the conventional GAN. The losses in the discriminator D and the generator G can be defined as:

$$V_{LSGAN}(G,D) = \frac{1}{2}E_{x\sim p_{data}(x)}[(D(x)-b)^2] + \frac{1}{2}E_{z\sim p_z(z)}[(D(G(z))-a)^2], \tag{3.22}$$

$$V_{LSGAN}(G) = \frac{1}{2}E_{z\sim p_z(z)}[(D(G(z))-c)^2]. \tag{3.23}$$

where c is the value that G hopes for D to believe for generated samples.

3.3.3 Wasserstein GAN

Since the real data distribution has less overlap with the generated data, JSD-based objective function can be a constant, which causes the vanishing gradient and training oscillation. Arjovsky *et al*. [103] proposed Wasserstein GAN (WGAN) by using the Earth-Mover distance (EMD) to replace JSD for measuring the distribution between the real data and the generated one. WGAN resorts to the Kantorovich-Rubinstein duality to define the loss function. WGAN made the progress of the training oscillation in GAN. The loss function of WGAN is shown as follows

$$V_{WGAN}(G,D) = E_{x\sim p_{data}(x)}[D(x)] + E_{z\sim p_z(z)}[D(G(z))], \tag{3.24}$$

where the discriminator D is optimized over the set of 1-Lipschitz functions. Informally, a 1-Lipschitz function is a differentiable function whose gradient has norm at most 1 everywhere. The adoption of such D in (3.24) allows for a smoother convergence relatively to (3.7).

However, it can still generate low-quality samples or fail to converge in some settings. In light of that, Gulrajani *et al*. [104] discovered that the training failures are often due to the use of weight clipping in WGAN to enforce a Lipschitz constraint on the discriminator, which can lead to pathological behaviour. Thus they propose an alternative method for enforcing the 1-Lipschitz constraint instead of clipping weights, penalizing the norm of the gradient of the discriminator concerning its input. Their method (clipping with gradient penalty, also known as WGAN-GP) converges faster and generates higher-quality samples than WGAN. The loss function of WGAN-GP is shown as

$$V_{WGAN-GP}(G,D) = E_{z\sim p_z(z)}[(D(G(z))] + \lambda_{gp} E_{z\sim p_z}(z)[(\nabla D(\alpha x - (1-\alpha G(z)))| - 1)^2]. \tag{3.25}$$

where α is a user-defined scaling factor and λ_{gp} stands for the gradient penalty coefficient and ∇D is the gradient of D.

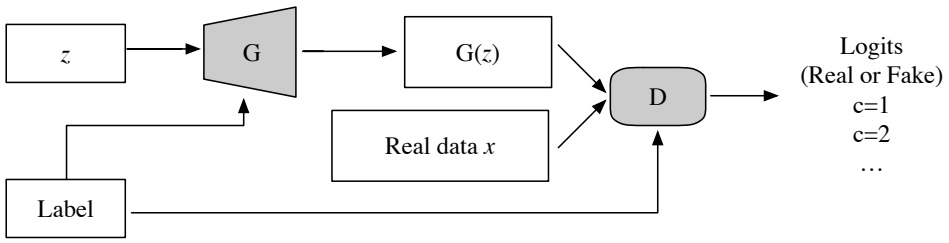


Figure 3.5: The architecture of conditional generative adversarial network (cGAN).

3.3.4 Conditional GAN

GAN can be extended to a conditional model if the discriminator D and the generator G are conditioned on some extra information y, i.e., labels, shown in Fig 3.5. The loss function of conditional GAN (cGAN) is

$$V_{CGAN}(G,D) = E_{x\sim p_{data}(x)}[logD(x|y)] + E_{z\sim p_z(z)}[log(1-D(G(z|y)))]. \tag{3.26}$$

where $D(x|y)$ and $G(z|y)$ represent conditional probabilities while y is the class label information.

Based on this formulation, the inputs are concatenated (embedded) with y to start training. Ultimately, we can generate desired samples by feeding corresponding y. The conditional information y can be formed like class labels [105], or text [106],[107],[108]. cGAN has been used for convolutional face generation [109], face editing [110], image translation [111], natural image description [112], 3D-aware scene manipulation [113] and machine translation [92].

3.3.5 Auxiliary classifier GAN

Odena *et al*. [114] proposed auxiliary classifier GAN (AC-GAN) for semi-supervised learning. The key to AC-GAN is that it can incorporate label information into the generator and adjust the loss function to be suitable for the discriminator. The difference between cGAN and AC-GAN is that cGAN's output is the probability of whether the image is real. For AC-GAN, the input to the discriminator is an image, while the output is the probability that the image is real and its class label. Thus, the loss function

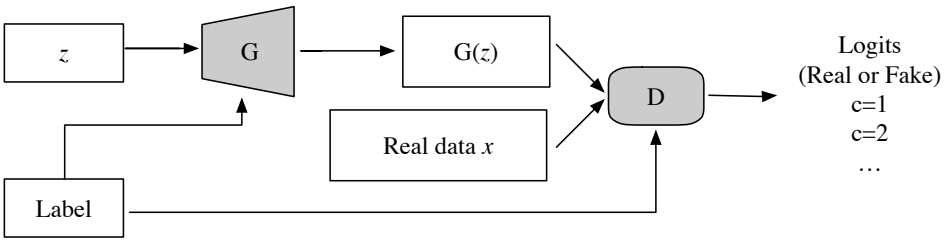


Figure 3.6: The architecture of AC-GAN.

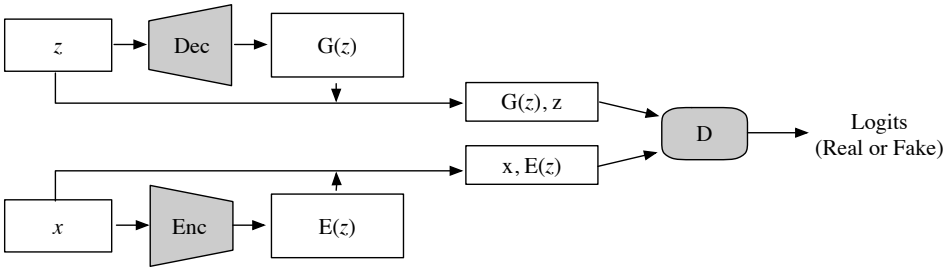


Figure 3.7: The architecture of Bi-GAN.

of AC-GAN is the same as cGAN but with a different framework shown in Fig. 3.6

3.3.6 Bidirectional GAN

Donahue *et al*. [115] raised a framework named bidirectional GAN (Bi-GAN) to map the real data to the latent space to achieve adversarial feature learning. Bi-GAN adds an extra encoder E on the basic GANs to map the real data x to latent space that the optimization problem is converted to $\min_{G,E} \max_{D} V(G,E,D)$. The framework of Bi-GAN is shown in Fig. 3.7 and its loss function is as follows.

$$V_{Bi-GAN}(G,E,D) = E_{x\sim p_{data}(x)}[logD(x,E(x))] + E_{z\sim p_z(z)}[log(1-D(G(z),z))]. \quad (3.27)$$

3.3.7 Cycle consistency GAN

The image-to-image translation is a research topic of interest in computer vision and image processing, where the objective is to learn the mapping relationship between output and input images using a set of aligned image pairs. However, reference [116] can not be used for unpaired data (no input/output pairs), which was well solved by cycle consistency GAN (CycleGAN) [33]; a promising approach for unpaired data.

The architecture of CycleGAN is shown in Fig. 3.8. In the first part, the real image

from domain A is sent to a generator (G12) to produce generated image of domain B. On the one hand, the generated domain B image and the real image of domain B will start adversarial learning. On the other hand, this generated image will be passed through another generator (G21) to reconstruct the image of the domain A. Both two processes learn together to form the first part of CycleGAN. The second part is the inverse version of the first part. These two parts are formed as CyclgeGAN. The loss function for CycleGAN is shown as

$$V_{CycleGAN}(G, D) = L_{adv}^{12} + L_{adv}^{21} + \lambda_c * L_{cyc} \tag{3.28}$$

where L_{adv}^{12} and L_{adv}^{21} are adversarial losses, L_{cyc} is the cycle consistent loss, λ_c is the correspond hyper-parameter. The specific formulations for these losses are as follows.

$$\begin{aligned} L_{adv}^{12} = & \mathop{\mathbb{E}}_{x \sim P_B}[\mathrm{D}_{12}(x_B)] + \mathop{\mathbb{E}}_{x \sim P_A}[(\mathrm{D}_{12}(\mathrm{G}_{12}(x_A))] \\ & + \lambda \mathop{\mathbb{E}}_{x \sim P_A}[(|\delta \mathrm{D}_{12}(\alpha_{12} x - (1 - \alpha_{12} \mathrm{G}_{12}(z)))| - 1)^2, \end{aligned} \tag{3.29}$$

$$\begin{aligned} L_{adv}^{21} = & \mathop{\mathbb{E}}_{x \sim P_A}[\mathrm{D}_{21}(x_A)] + \mathop{\mathbb{E}}_{x \sim P_B}[(\mathrm{D}_{21}(\mathrm{G}_{21}(x_B))] \\ & + \lambda \mathop{\mathbb{E}}_{x \sim P_B}[(|\delta \mathrm{D}_{21}(\alpha_{21} x - (1 - \alpha_{21} \mathrm{G}_{21}(z)))| - 1)^2, \end{aligned} \tag{3.30}$$

$$\begin{aligned} L_{cyc} = & E_{x_A}[||x_A - \mathrm{G}_{21}(\mathrm{G}_{12}(x_A))||_2] \\ & + E_{x_B}[||x_B - \mathrm{G}_{12}(\mathrm{G}_{21}(x_B))||_2]. \end{aligned} \tag{3.31}$$

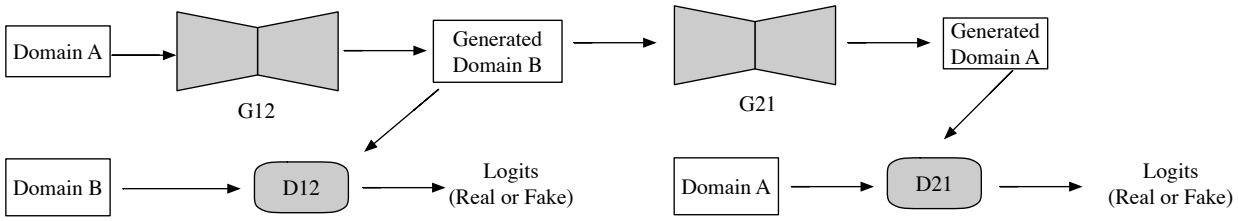


Figure 3.8: The architecture of CycleGAN.

3.3.8 Auto-encoder with GAN

The auto-encoder (AE) is an ANN that learns to compress data. Briefly, we introduce three layers of AE (it can be more layers in the hidden layer) shown in Fig. 3.9. Suppose given the input vector $x = \{x_1, x_2, x_3, x_4\}$, it will go through the encoder Enc to the latent space of $z = \{z_1, z_2, z_3\}$, and then it will reconstruct z to $\bar{x} = \{\bar{x}_1, \bar{x}_2, \bar{x}_3\}$ in the decoder Dec. Generally, the mapping process can be shown as

$$z = Enc(x), \tag{3.32}$$

$$\bar{x} = Dec(z). \tag{3.33}$$

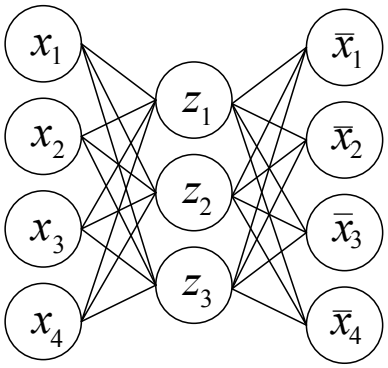


Figure 3.9: The architecture of auto-encoder: x represents the input neurons, z stands the neurons in the latent space and $\bar{x}$ denotes the reconstructed x. See text for details.

However, the objective of compressing data is not only to reduce the number of dimensions of the data but to keep the major part of the data structure information in the reduced representation. In AE, the number of dimensions in the reduced representation should be carefully adjusted to suit the regularity of the latent space. In the generation process, the latent vector in AE may not be regular enough to get the desired output. Besides, with high-dimensional data for compressing in AE, will lead to overfitting, implying that some points of the latent space will give meaningless content once decoded. To this end, the Variational auto-encoder (VAE) is introduced which has the same neural network architecture as AE. In VAE, the encoding distribution is regularised during the training to avoid overfitting and can ensure that the latent space

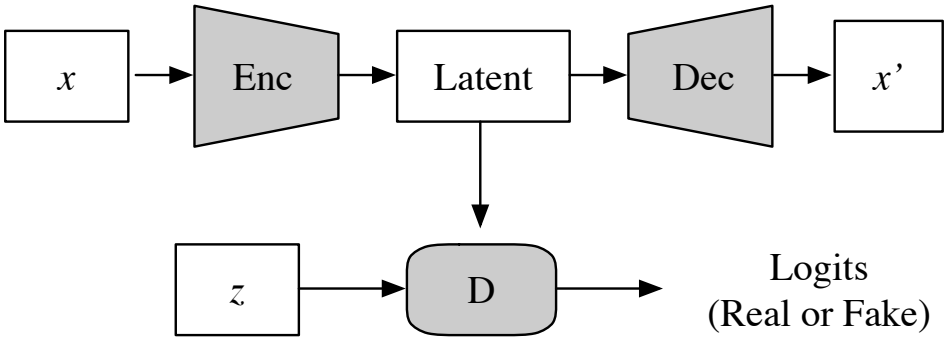


Figure 3.10: The architecture of AAE.

has good properties to generate data, which has the main data structure information. In VAE, *KL*-divergence measures the differences between two distributions, shrinking the encoded latent variables into a normal distribution.

The adversarial auto-encoder (AAE) in [117] is a AE based on GANs. It is an interesting framework for blending AE architecture with the adversarial concept. It uses a similar concept to VAE, except that it uses adversarial loss to regularize the latent code instead of the KL-divergence. The encoder of AAE will act as the generator. It uses an adversarial loss where an additional discriminator component is added to force the model to learn more realistic distributions. Unlike GAN, where the output of the generator is the generated samples, and the input for the discriminator is both the real and fake samples, AAE's generator (Enc in Fig. 3.10) generates a latent code and tries to fool the discriminator into believing that the latent code is sampled from the chosen distribution. The discriminator will automatically evaluate whether a given latent variable is generated by the auto-encoder (fake) or a random vector sampled from the normal distribution (real). AAE is a different learning mechanism from the conventional GAN that takes an image as an input instead of the random distribution. The framework of AAE is shown in Fig. 3.10.

3.4 Evaluation metrics for GAN

Another issue in GAN design is the metric for selecting the best generator. Due to the adversarial learning, the convergence curve oscillates and thus the generator in the

last iteration is hardly the best one. Thus, designing a metric for generator selection is significant for enhancing the generation performance of GAN. In this section, we show some evaluation metrics for keeping track of the best generator during training.

3.4.1 Inception score

The inception score (IS) [118] uses for every generated sample to get the conditional label distribution $p(y|x)$. It is an automatic evaluation of the quality of generated images. The formula for IS is shown as

$$IS = \exp^{(E_x KL(p(y|x)||p(y)))}. \tag{3.34}$$

3.4.2 Mode score

The mode score (MS)[119, 120] is an improved IS. The difference is that MS can measure the dissimilarity between the real distribution and generated distribution.

3.4.3 Fréchet Inception distance

The Fréchet Inception distance (FID) [121] is another metric to estimate the performance of GANs. Given two distributions p_{data} from real samples and p_g from the generator, the corresponding formula is shown as

$$FID(p_{data}, p_g) = ||\mu_r - \mu_g|| + tr(C_r + C_g - 2(C_r C_g)^{\frac{1}{2}}), \tag{3.35}$$

where C_r and C_g are empirical covariance.

3.4.4 Multi-scale structural similarity (MS-SSIM)

The Multi-scale structural similarity (MS-SSIM) [122] is designed to capture the similarity between two samples (the real samples and the generated one). It quantitatively evaluates samples' similarity by attempting to predict human perceptual similarity

judgment. The corresponding formula for calculating the similarity is shown as

$$SSIM(x,y) = f(l(x,y),c(x,y),s(x,y)) \tag{3.36}$$

where $l(x,y)$, $c(x,y)$ and $s(x,y)$ are well detailed in [122].

How to select a good evaluation metric for GAN is still a challenging problem [123]. Hopefully, there will be better methods for evaluating the generative quality of GAN models in the future.

3.5 Application of GAN

As discussed, GAN is a powerful tool to generate real-like samples. Besides, GAN has been widely used for speech and machine translation. In this section, we introduce some applications of GAN.

3.5.1 Image and computer vision

The most successful applications scenarios of GAN are in image processing and computer vision, such as high-resolution image synthesis and video processing. Due to its outstanding performance in generating images, Wang *et al*. [124] introduced a DCGAN architecture method for synthesizing high-resolution photo-realistic images. Results demonstrate that this framework can generate more diverse of the generated images given the same input images, allowing users to edit the objective appearance. Odena *et al*. [125] constructed a model based on AC-GAN for image synthesis. Results showed that this new method could improve the training of GANs for image synthesis. Ledig *et al*. [31] used GAN for image super-resolution (SRGAN). In the experiment, the VGG network is applied in the discriminator, and the residual network is used in the generator. Experimental results show that SRGAN can obtain rich texture details for estimating realistic super-resolution images. Ghosh *et al*. [126] argued that recent progress in GAN cannot provide an inference model for image editing or classification.

Figure 3.11: Face samples generated by Progressive GAN [2].

Besides, these methods are not useful for extending to novel datasets or architectures. Under these two issues, they proposed a general framework called Invertible GANs (InvGAN) that are agnostic to architectures and datasets. Qualitative and quantitative experiments showed that InvGAN can successfully transform image in painting, merging, interpolation, and on-line data augmentation.

3.5.2 Machine translation

The objective of machine translation is to translate one language to another as performed through software and unaided by the human hand. However, since the data are discrete, it is difficult for GAN to generate such text samples. GAN will score the entire text sequence when generating text sequences. For a partially generated sequence, it is tough to judge its score when generating a complete sequence later.

To migrate this problem and successfully generate synthetic text, in 2016, Zhang *et al*. [127] proposed a generic framework employing long short-term memory and CNN for adversarial training to generate realistic text. The method adopts an objective function similar to feature matching [36] instead of the loss function of conventional GAN. Experimental results demonstrate that the proposed model can generate realistic sentences via adversarial training. Since it is difficult to pass the gradient update from the discriminator to the generator when producing discrete output during adver-

sarial training, Yu *et al*. [128] developed a sequence GAN framework named SeqGAN for modeling sequence data. The method is based on a process of sequential decision-making and reinforced learning (RL). Experiments conclude that SeqGAN can significantly improve the quality of synthesis sequence data. In 2017, Li *et al*. [129] used the adversarial learning idea for open-domain dialogue generation. The method also processed RL problems with GANs like SeqGAN. The experimental results show that the proposed method with an adversarial training mechanism can generate higher-quality responses. Pfau *et al*. [130] applied a method named actor-critic methods with GAN for sequence prediction.

3.5.3 Industrial machinery fault diagnosis

Typically, the objective of a deep learning-based fault diagnosis is automatic feature generation. It played a significant role in the machinery fault diagnosis. In these tasks, data acquisition is performed by sensors such as accelerometers and attitude sensor. in general, labelled data are required. However, acquiring labelled data from equipment operating under faulty conditions is relatively more complicated than acquiring data from healthy states. Some faulty conditions are not sustainable and may lead to a machine operation interruption before sufficient data can be acquired. As a consequence, an imbalanced data set is all one can acquired. The imbalanced data set can lead to poor model performance, especially in deep learning, where large quantities of data are required.

To mitigate this problem, some researchers studied data augmentation techniques based on GANs. Pu *et al*. [35] investigated GANs as an oversampling tool for generating a synthetic data set for an industrial robot fault classification. Cabrera *et al*. [131] proposed a generator selection method for GANs under extremely imbalanced data for machinery condition monitoring. Mao *et al*. [86] used GANs for an imbalanced data-driven fault diagnosis of rolling bearings. Also, for rolling bearings, Jiang *et al*. [88] designed a novel anomaly detection approach based on GANs with only health data. Li *et al*. [89] applied GANs for the feature space learning in fault diagnosis of 3D print-

ers using only one sample of each faulty state. Wang *et al*. [90] proposed a technique based on a conditional VAE and GANs for imbalanced fault diagnosis of planetary gearboxes. Li *et al*. [132] proposed a novel deep learning on CNN for rotating machinery fault diagnosis with limited labeled data. Peng *et al*. [133] presented an improved SMOTE for data set optimization and then used an optimized k-nearest neighbour for wind turbine blade fault classification. Shao *et al*. [134] developed an AC-GAN based framework to produce enough data for fault diagnosis and test its performance on an induction motor fault simulator. Han *et al*. [135] designed a novel deep adversarial CNN for intelligent fault diagnosis of mechanical faults.

3.5.4 Other applications

Enterprises are interested in modeling natural and commercial phenomena. Since modeling financial time series is a challenge for GAN, Wiese *et al*. [136] developed a quant GAN for financial time-series modelling. To assist with classifying credit card fraudulent transactions, Ba *et al*. [137] applied GAN to generate synthesis data for credit card fraud detection. In the medical area, Wu *et al*. [138] used medical generative adversarial network (MedGAN) for generating of screening cancer images.

3.6 Case study: GAN as an oversampling method for data augmentation in an industrial robot fault diagnosis task

3.6.1 Introduction

Data-driven machine learning techniques play an important role in the machinery fault diagnosis and prognostic [139, 140, 141]. Recently, deep learning [142] emerged as one that has been progressively adopted to develop health monitoring systems for the machinery. One way of looking at deep learning is as a feature engineering method [143]

that automatically extracts features from the collected signals. Propagating these signals from the input layer to layers with fewer and fewer neurons, the neural network is forced to represent the input data space into a lower dimensional feature space which, in general, reduces overfitting and increases the accuracy.

Deep learning models such as CNN [144], deep AE [145], DBN [146] and DBM [147] have achieved outstanding results in fault diagnosis and other fields essentially due to the dimensionality reduction effect. In the industrial robot field, Nho Cho *et al*. [148] proposed an algorithm based on MLP for the robot actuator fault detection. Wang, *et al*. [149] proposed a multi-data fusion with an optimized CNN for fault detection of rotating machinery. Ma *et al*. [150] proposed the convolutional multi-time scale ESN for efficient classification. Three different ESNs with different time scales were used to allow the recurrent neural network to successively refine the features in a similar way to a kernel in a CNN. Hu *et al*. [151] presented an approach with DBM and multi-grained scanning forest to effectively deal with industrial fault diagnosis. Wang *et al*. [152] proposed a new deep ANN model based on a DBM for condition prognosis. Lee *et al*. [153] proposed a real-time fault diagnosis model using a deep ANN. Shao, *et al*. [154] presented a continuous DBN for bearings fault detection. Shen *et al*. [146] used a DBN with an optimized function of Nesterov momentum for bearing fault diagnosis. Polic *et al*. [155] presented a new method of a CNN-based encoder for feature extraction in tactile robotics. D'Elia *et al*. [156] based on the study of how the power flows inside the time synchronous average of the ring gear and a modified statistical parameter for planet gears fault diagnosis. Zaidan *et al*. [157] used a Bayesian hierarchical model that utilizes fleet data from multiple assets to perform probabilistic estimation of remaining useful life for civil aerospace gas turbine engines.

However, all the above models are critically dependent on a representative, balanced, large enough data set that require data that, more often than not, are hard to obtain [158]. While data from a healthy state are abundant, data from faulty states are rare, sparse, and hardly representative of all possible faults. This could lead to low diagnosis precision in these intelligent fault diagnosis techniques. Without appropri-

ate data, these machine learning methods simply do not have acceptable performance [159]. It is important to consider a method that mitigates such a data shortage.

Robots are wildly used in the industry as they can be used in a range of tasks such as assembly, painting or welding [160]. However, the transmission system of the robot is prone to faults due to prolonged working periods [161]. Typically, these faults manifest in the connection parts, bearings, gears, or gear shafts. A faulty robot will be less precise, less efficient, less productive, and less secure. Even though, it could be a tricky task to obtain fault data for such precision machinery. Therefore, the monitoring of the robot health condition with limited data sources is of paramount interest.

For mitigating this, some fault diagnosis works resorted to the SMOTE. SMOTE stands for Synthetic Minority Oversampling Technique and aims at compensating for the data unbalance of a given class by increasing the number of samples in that class. Roughly, it creates a new synthetic sample by interpolating two existing similar samples of the same class, the new sample has the same class of the two originating samples [162]. SMOTE is a popular method, but sometimes, by increasing the number of samples, it will also increase the overlapping between classes. An alternative approach to cope with the imbalanced data set problem is to resort to deep learning and in particular to GANs. The idea is to use the generative model of GANs to generate enough samples for effective training of the diagnoser.

In the remaining of this section, the proposed GANs-based fault diagnosis scheme is specified in Section 3.6.2, including feature extraction using Wavelet packet transform, the theoretical background of random forest for fault classification. In section 3.6.3, the considered scenarios and the hyper-parameter settings are detailed. In section 3.6.4, we provide a complete analysis under the fault diagnosis of an industrial robot.

3.6.2 Methodology

(a) Feature extraction

Wavelet packet transform (WPT) can be viewed as a time-frequency conversion technique of a non-stationary signal [163]. It complements the shortage of wavelet transforms that only decomposes the low frequency components but cannot extract high-resolution on high frequency components. The discrete wavelet transform of the discrete signal $f(t)$ is given by [164]:

$$w_{m,n}(t) =< f(t), \Psi_{m,n}(t) >= \frac{1}{\sqrt{m}} \int_{-\infty}^{+\infty} \Psi_{m,n}(\frac{t-n}{m}) f(t) dt, \tag{3.37}$$

where m is the scaling factor and n is the sifting factor which are given, respectively, by:

$$m = a_0^m, \tag{3.38}$$

$$n = b_0 a_0^m. \tag{3.39}$$

When $a_0 = 2, b_0 = 1$, (3.37) can be re-written as

$$\Psi_{m,n}(t) = 2^{-\frac{m}{2}} \Psi(2^{-m}t - n) \tag{3.40}$$

In the wavelet transform, the signal $u(t)$ can be separated in the Hilbert space by scaling and by a wavelet function. The scaling function $\Phi(t)$ corresponds to the low frequency part of the original signal while the wavelet function $\varphi(k)$ corresponds to the high frequency part of the original signal with initial conditions:

$$\Psi_{m,n}^0(t) = \Phi(t) \tag{3.41}$$

$$\Psi_{m,n}^1(t) = \varphi(t) \tag{3.42}$$

Fig. 3.12 shows a 3-level decomposition of a WPT of a signal $u(t)$. This is decomposed in a high-frequency part $h_k(t)$ and in a low-frequency part $g_k(t)$. Each part is computed

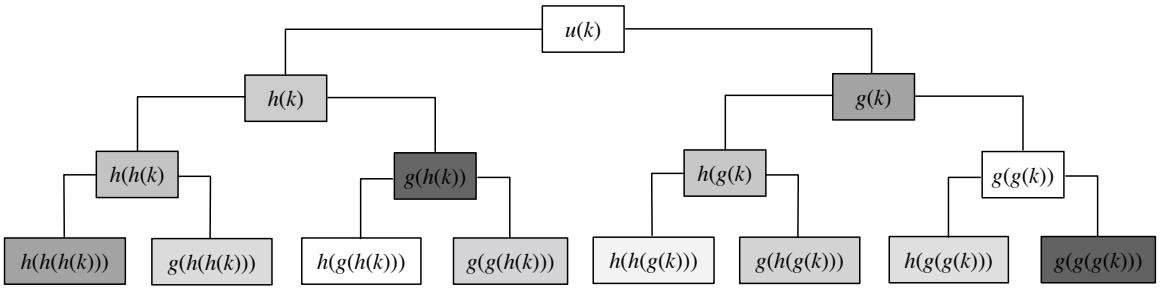


Figure 3.12: The decomposition levels of a wavelet packet transform of the signal $u(t)$.

by a filter, i.e.,

$$h_k(t) = \frac{t_k + t_{k+1}}{2}, \tag{3.43}$$

$$g_k(t) = \frac{t_k - t_{k+1}}{2}, \tag{3.44}$$

The function using the above filters can be given by:

$$u_{2n}(t) = \sum_k h_k u_n(2t - k), \tag{3.45}$$

$$u_{2n+1}(t) = \sum_k g_k u_n(2t - k) \tag{3.46}$$

As illustrated in the figure, this procedure can be recursively applied to both low and high-frequency parts. However, the number of decomposition levels will be limited by the actual application. Due to its smoothness and non-linear characteristics, in this paper, we applied the Daubechies WPT with 7 levels (Db7).

We further compute an informative statistics from the WPT [165] as follows:

$$p(m) = \sqrt{\sum_{n=1}^{N} (w_{m,n}(t))^2} \tag{3.47}$$

where N denotes the number of data in each node of the 7th decomposition level and $w_{m,n}(t)$ and is given by (3.37). Hence, a feature vector $\mathbf{p}$ can be defined for the signal $u(t)$ as follows:

$$u(t) \leftrightarrow \mathbf{p} = [p(1), p(2), \ldots, p(d)]^{\mathrm{T}} \tag{3.48}$$

where d is the number of features, i.e., with 7 levels $d = 2^7 = 128$.

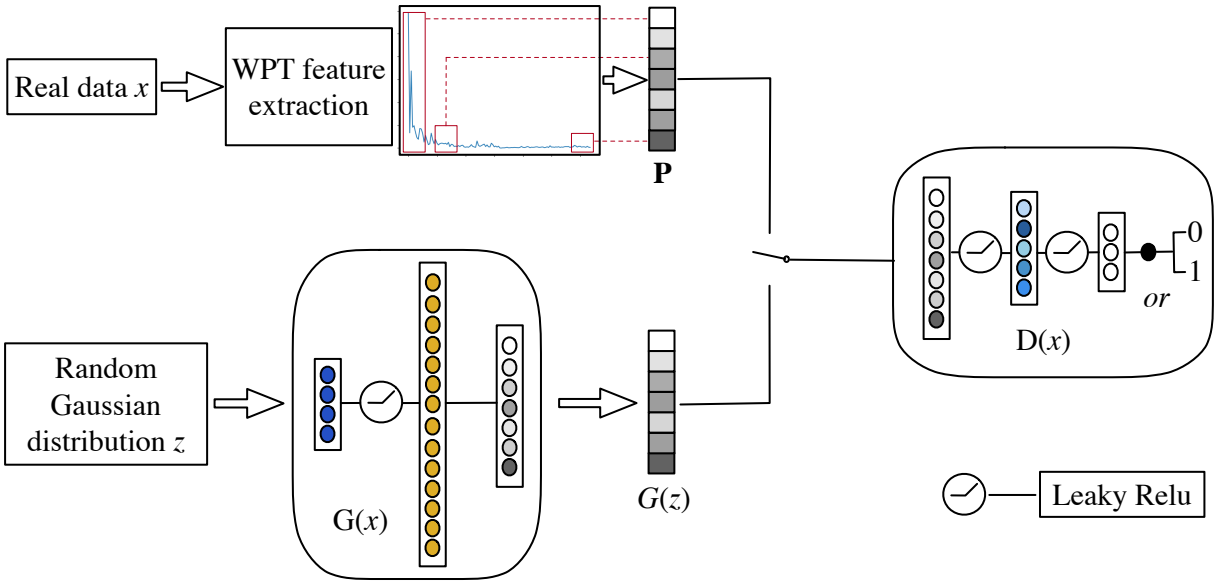


Figure 3.13: The learning scheme of the fault diagnoser.

With this in mind, Fig. 3.13 shows a block diagram of GAN as used in this work. For learning, GAN implements an adversarial competition between the generator G(z) and the discriminator D(x). Initially, a real sample $u(t)$ is processed by the above described WPT feature extraction technique, and (3.48) is set as the real input of D(x). A random signal $z(n)$ with a given distribution is inputted to the generator, which in turn produces a synthetic feature vector G(z). The discriminator is trained with a target value of 1 when a real sample is presented at its input and 0 for a synthetic example. This process repeats until a Nash equilibrium is reached.

(b) Keeping track of the best generator

Cabrera *et al*. [131] proposed a stop criterion aiming at keeping track of the best current generator while training progresses. The metric is given by:

$$||\mathrm{D}_r - \mathrm{D}_g|| = (R_{mean} - G_{mean})^2 + (R_{std} - G_{std})^2 \tag{3.49}$$

where R_{mean} and G_{mean} are the centroids of the real and generated data clusters, respectively, while R_{std} and G_{std} are the real and generated data dispersion (standard deviation), respectively.

The smaller (3.49), the closer the generated data is to real data. In each training step,

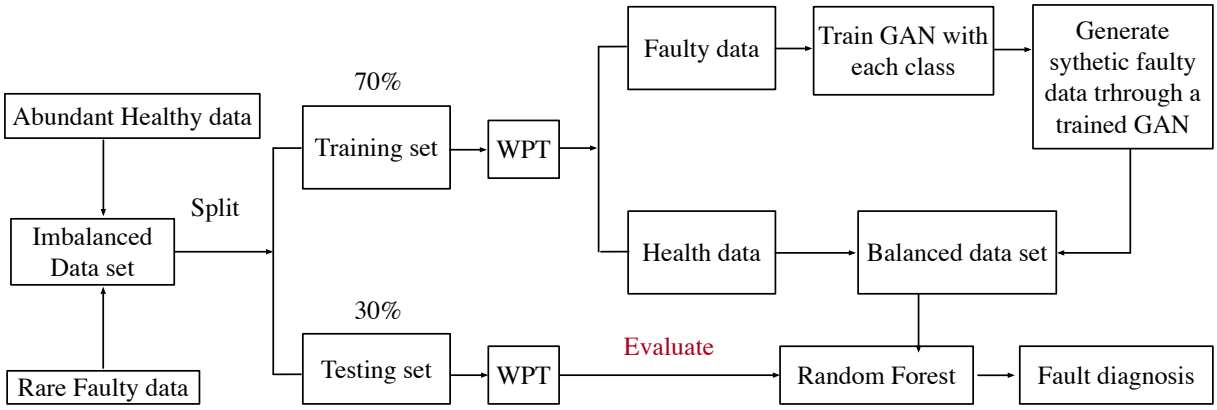


Figure 3.14: The flow chart of the procedure of the proposed approach.

(3.49) is computed for the current generator and the generator exhibiting the lower current distance is viewed as the best current generator. Hereafter we refer to this process as (model) generator selection.

(c) Data generation for fault classification

Based on the feature extraction process that uses WPT, illustrated in subsection 3.6.2 (a) and an RF classifier detailed in the previous section, one can be set up the learning scheme for the robot fault diagnoser. As shown in Fig. 3.13, a real data observation of from each class is sent to the WPT to extract the vector of features (3.48). Meanwhile, a random signal z is input into the generator that will produce a vector of synthetic features $\mathrm{G}(z)$.

The goal of the discriminator $\mathrm{D}(x)$ is to distinguish between the real vector of features, outputting a 1, and the synthetic vector of features, outputting a 0. The learning process described in section 3.2 is applied. Once both the generator and discriminator are trained, the generator can be used to generate as much synthetic data as required. Notice that the learning process and subsequent synthetic generation are carried out for each faulty class, i.e., for each class, we need to increase the number of observations. Finally, (health and faulty) real data together with faulty generated data are used in the random forest classifier for fault classification. Based on the above description, the procedure can be outlined in the following flow chart (Fig. 3.14). Besides, this whole process is illustrated in Fig. 3.15.

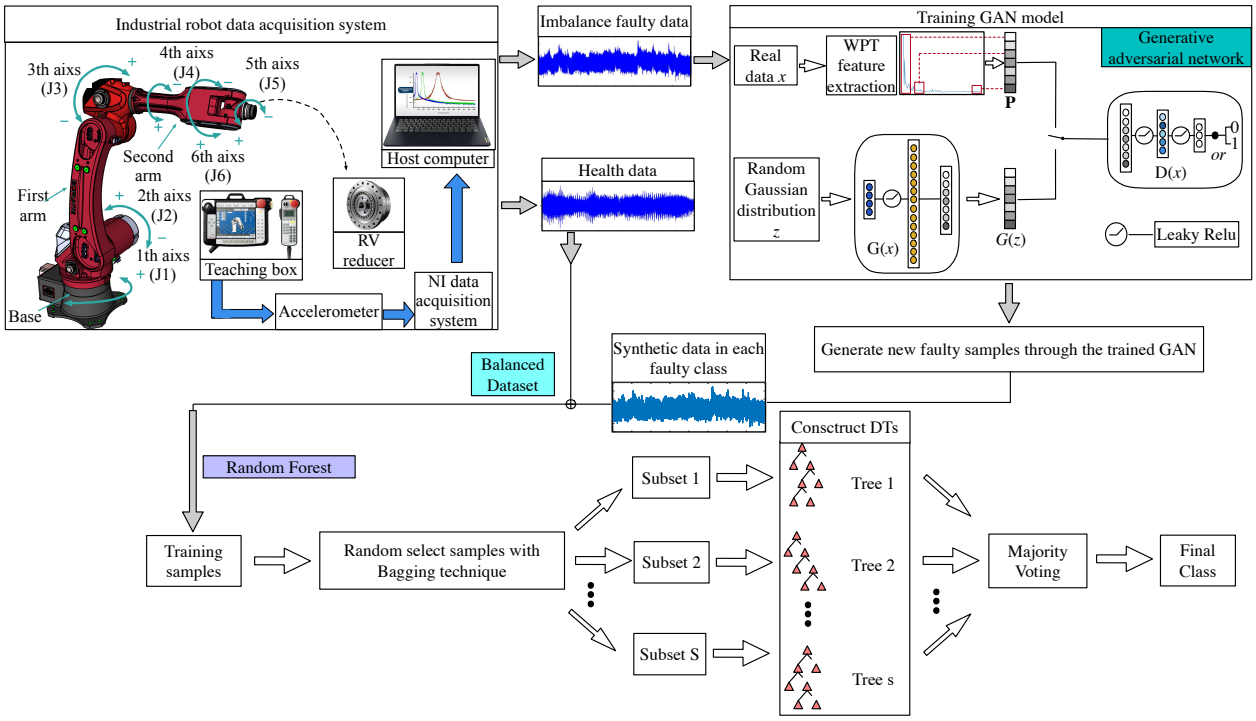


Figure 3.15: The complete data pipeline for fault diagnosis of the manipulator.

3.6.3 Experiments

(a) Considered scenarios

Several scenarios were considered to assess the effectiveness of the proposed model. In all these scenarios, we all used an RF with 1000 trees for classification. The different scenarios are identified as follows: RF-i denotes a random forest trained with the imbalanced dataset described in Table 2.2; RF-b2 denotes a random forest trained with a subsampled balanced dataset with only 140 observations per condition; RF-GAN and RF-GAN2 stand for rand forests trained with data sets that have the real 14,000 healthy observations and 14,000 faulty observations, the only difference between these scenarios is that RF-GAN uses the technique described in Section 3.6.2 to select the best current model for generating samples while RF-GAN2 uses the model obtained in the last iteration (i.e., iteration 10 000) of the training process to generate the samples. RF-GAN1 is similar to RF-GAN with the difference that only 13860 synthetic faulty states were generated while the remaining 140 are the original faulty states presented in the training data set. For comparison purposes, we also considered a random forest trained with a data set previously processed by SMOTE, a popular oversampling

method for dealing with imbalanced data sets.

(b) Hyper-parameter settings

In all GAN-based models, the generators are MLP with a 64:1014:128 fully connected topology while discriminators are also MLP with a 128:1024:2048:1 fully connected topology. These were selected empirically after some preliminary tests. The Adam optimizer is used with its key parameter settings of β_1=0.9, β_2=0.999 and ϵ=1e-08. The learning rate for the generators is set to 1e-5 while for the discriminators is set to 1e-4. The α and λ are set to 1e-4 and 1.0, respectively. A maximum number of 10,000 iterations was set for training.

3.6.4 Results and discussion

(a) On different scenarios for dealing with the imbalanced data set

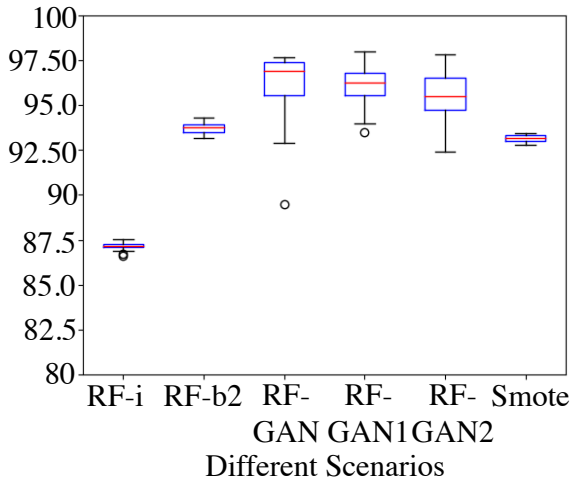


Figure 3.16: Boxplots exhibiting the relative distributions of accuracy obtained with the different scenarios considered for fault classification. See text for details.

Fig. 3.16 shows the distribution of the obtained accuracy for each scenario using boxplots. A boxplot summarizes a data distribution stressing five characteristic values: minimum, lower quartile, median, upper quartile, and maximum value. The red line denotes the median value. The distribution pertains to 20 independent repetitions.

The results presented in Fig. 3.16 were analyzed by the Friedman test, a non-parametric statistic test of hypotheses, to evaluate whether or not there is a statistically significant difference between the results of the different scenarios. The Friedman null hypothesis is that there is no statistically significant difference between the results of the different scenarios. Given a significant level, α, this hypothesis cannot be rejected whenever the p_{Friedman}, the p-value generated by the test, satisfies $p_{\text{Friedman}} > \alpha$. The null hypothesis is rejected otherwise, meaning that there is a statistically significant difference between the analyzed scenarios. In such a case, we can detect which of the scenario is responsible for such a difference by resorting to a pairwise *posthoc* test. A ranking can be obtained by counting the number of times that a method was a winner in the pairwise comparison. See [166, 167] for further details. Here we use the usual $\alpha = 0.05$ and the Wilcoxon test as *posthoc*.

Table 3.1
Wilcoxon *posthoc* pair-wise tests for the different scenarios.

Pair	p**-value**	**Winner**
RF-i vs. RF-b2	8.857E-05	RF-b2
RF-i vs. RF-GAN	8.857E-05	RF-GAN
RF-i vs. RF-GAN1	8.857E-05	RF-GAN1
RF-i vs. RF-GAN2	8.857E-05	RF-GAN2
RF-i vs. SMOTE	8.857E-05	SMOTE
RF-b2 vs. RF-GAN	8.857E-05	RF-GAN
RF-b2 vs. RF-GAN1	8.857E-05	RF-GAN1
RF-b2 vs. RF-GAN2	8.857E-05	RF-GAN2
RF-b2 vs. SMOTE	1.204E-04	SMOTE
RF-GAN vs. RF-GAN1	0.079	–
RF-GAN vs. RF-GAN2	8.857E-05	RF-GAN
RF-GAN vs. SMOTE	8.857E-05	RF-GAN
RF-GAN1 vs. RF-GAN2	8.845E-05	RF-GAN1
RF-GAN1 vs. SMOTE	8.844E-05	RF-GAN1
		RF-GAN / RF-GAN1

When Friedman was applied to the results in Fig. 3.16 yielded $p_{\text{Friedman}} = 1.865E-19 < 0.05$ meaning there is a statistically significant difference between the six scenarios. Table 3.1 shows the subsequent *posthoc* results. From these, one should conclude that there is no statistically significant difference between scenarios RF-GAN and RF-GAN1 and that these outperform all the others. This is an interesting observation as both RF-GAN and RF-GAN1 use the technique described in Section 3.6.2 (b) to select

the best current generator and that the only difference between these scenarios is that in RF-GAN all the faulty state data are synthetic while in RF-GAN1 only 13860 faulty examples are synthetic while the remaining 140 are the original faulty states presented in the training data set. This further endorses the quality of the obtained generators.

The average accuracy of RF-i was 87.88% while with an undersampling balanced dataset, RF-b2 reached 94.5%. RF-GAN had an average accuracy of 97.06%, RF-GAN1 97.75%, and RF-GAN2 95.38%. SMOTE had an average accuracy of 95.17% . We observe that the GANs-based average accuracies are all higher than the imbalanced (RF-i), undersampling (RF-b2), and oversampling (SMOTE) scenarios. Within the GANs-based scenarios, RF-GAN has shown a difference of 1.68% relative to RF-GAN2 in average accuracy.

(b) On of the performance in each class

To further analyze the above results, the recall indicator of each fault class is studied. As it is shown in Fig. 3.17, the recall indicator in the RF-i model of the health state reaches 100% while for the other 3 faulty classes comes down to 63.88% in gear pitting; 84.3% in gear broken tooth, and 63.35% in gear cracking for an average of 77.88%. This clearly shows the effect of the imbalanced data set. For RF-GAN the recall indicator in each class is 99.53%, 99.73%, 99.63% and 99.87%, respectively. In RF-GAN1, the recall indicator in each class is 98.63%, 98.67%, 98.28%, and 95.40%, respectively. For the RF-GAN2 model, the recall indicator in each class is 99.30%, 93.08%, 98.15%, and 90.08%, respectively. The high recall indicators are due to sufficient examples in each class. This is also visible in the recall of SMOTE.

The fit score is another metric that can be used to compare the relative performance of the different scenarios in each class. From Fig. 3.18, one can see that the performance of RF-i as measured by the F1-score is 70.38% in fault A, 77.54% in fault B, 90.03% in fault C, and 76.49% in fault D. The performance of RF-b2 in each faulty condition is 91.05% in fault A, 92.25% in fault B, 96.50% in fault C and 94.3% in fault D respectively. The performance of RF-GAN is 98.52% in fault A, 98.83% in fault B, 95.66% in fault C

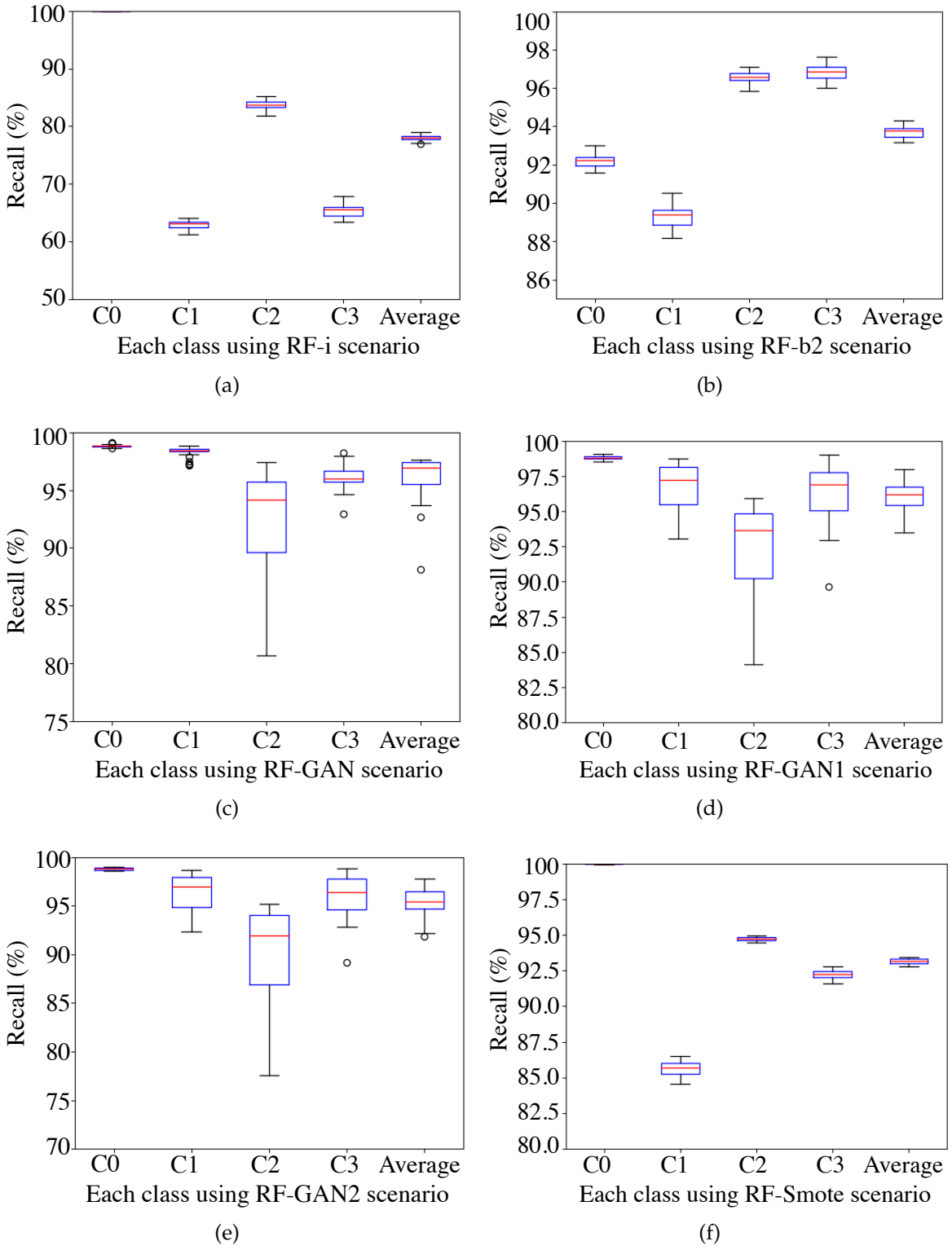


Figure 3.17: Recall indicators for the different scenarios: (a) RF-i; and (b) RF-b2; (c) RF-GAN; (d) RF-GAN1; (e) RF-GAN2; and (f) SMOTE.

and 95.23 in fault D. The performance of RF-GAN1 is 98.55% in fault A, 98.95% in fault B, 96.87% in fault C and 96.62 in fault D. The performance of RF-GAN2 is 95.70% in fault A, 96.54% in fault B, 90.01% in fault C, and 94.57% in fault D. For SMOTE, the

perform is 89.72% in fault A, 93.08% in fault B, 97.56% in fault C, and 95.28% in fault D.

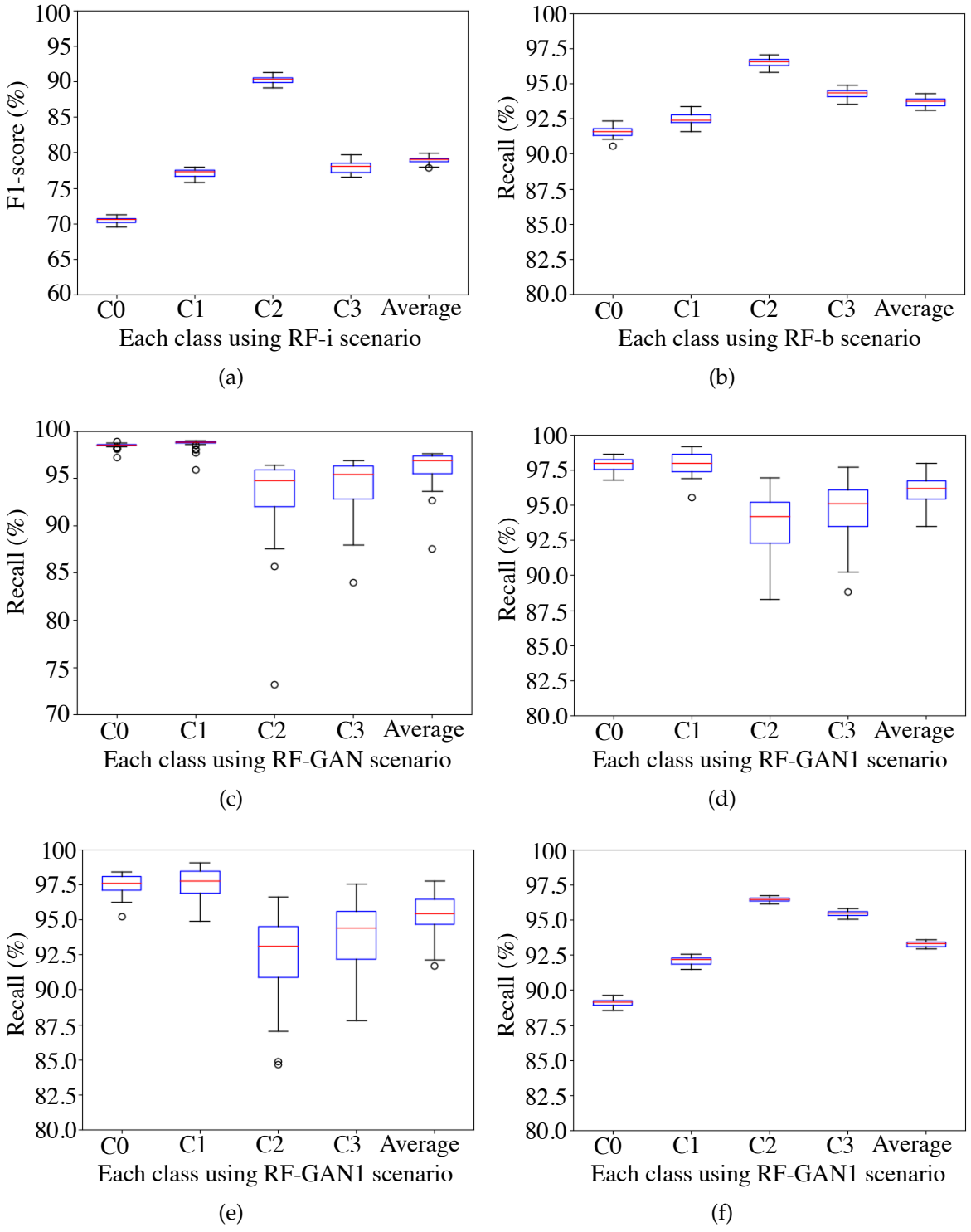


Figure 3.18: F1-score for scenario: (a) RF-i; and (b) RF-b2; (c) RF-GAN; (d) RF-GAN1; (e) RF-GAN2; and (f) SMOTE.

For completeness, the confusion matrices are presented in Fig. 3.19. All these ma-

trices consider the 6000 test observations as per Table 2.2. For RF-i (Fig. 3.19 (a)), one can see a high number of misclassifications in non-healthy states due to the imbalanced training set. These misclassifications are strongly reduced (especially in the GAN-based scenarios) when enough data is generated and used for training.

(c) On the training set

Learning curves: The performance of the proposed model under different data set sizes is now considered. Fig. 3.20 shows the average performance over 20 independent runs in the testing set for the scenario RF-GAN when only a given percentage of faulty observations are available for training. More concretely, the following percentages were considered i =1, 2, 4, 6, 8, 10, 20, 40, 60, 80, and 100 %. For instance, when $i = 4\%$, the number of training examples in each faulty state is $14000 \times 0,04 = 560$. It can be seen that the average accuracy increased from 56.25% to 97.05% by increasing the availability of faulty data from 1% (140 examples) to 100 % (14000 examples) in each fault type. There was a strong increase in performance up to 20%. After that point the improvement in accuracy was slower and slower until about 80%. After this value, the improvement was neglectable. That is, adding more data after a certain point hardly improves the performance.

Shuffling data: When generating training examples from a GAN-based model, shuffling the training data is a key important factor for obtaining an acceptable performance. Fig. 3.21 illustrates the importance of data shuffling. The results presented in this figure were obtained with exactly the same RF-GAN configuration, the only difference being the way data is presented to the GAN for training. For non-shuffled data, the model is simply not able to generate no matter the other initial conditions (weights).

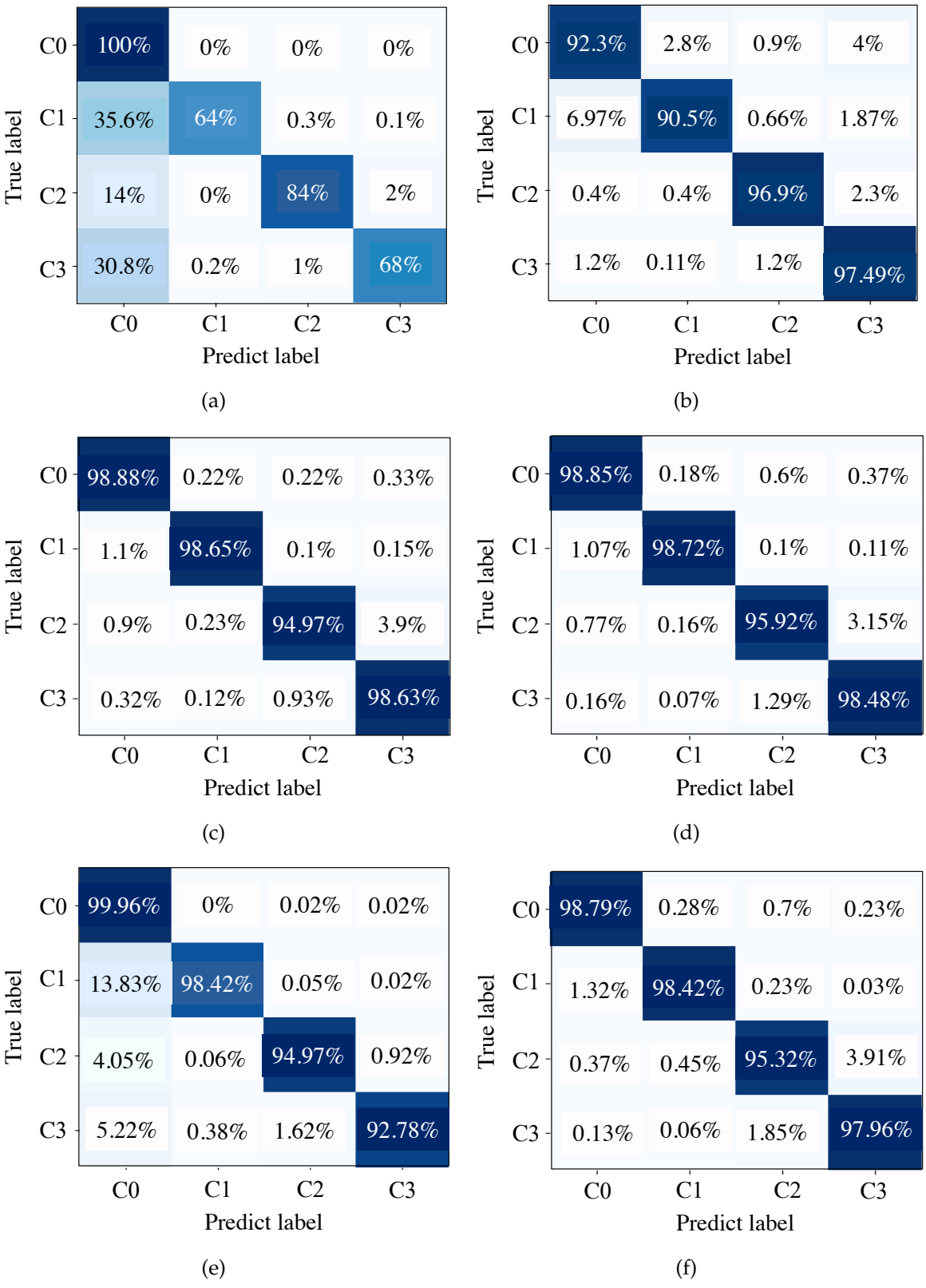


Figure 3.19: The confusion matrix for: scenario: (a) RF-i; and (b) RF-b2; (c) RF-GAN; (d) RF-GAN1; (e) RF-GAN2; and (f) SMOTE.

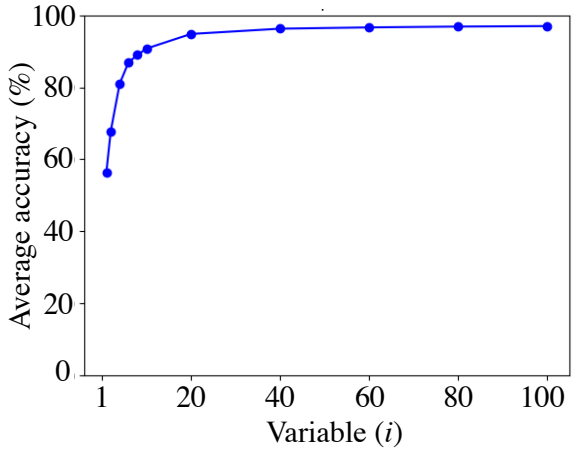


Figure 3.20: Learning curve of scenario RF-GAN for i =1, 2, 4, 6, 8, 10, 20, 40, 60, 80, and 100% of the training set.

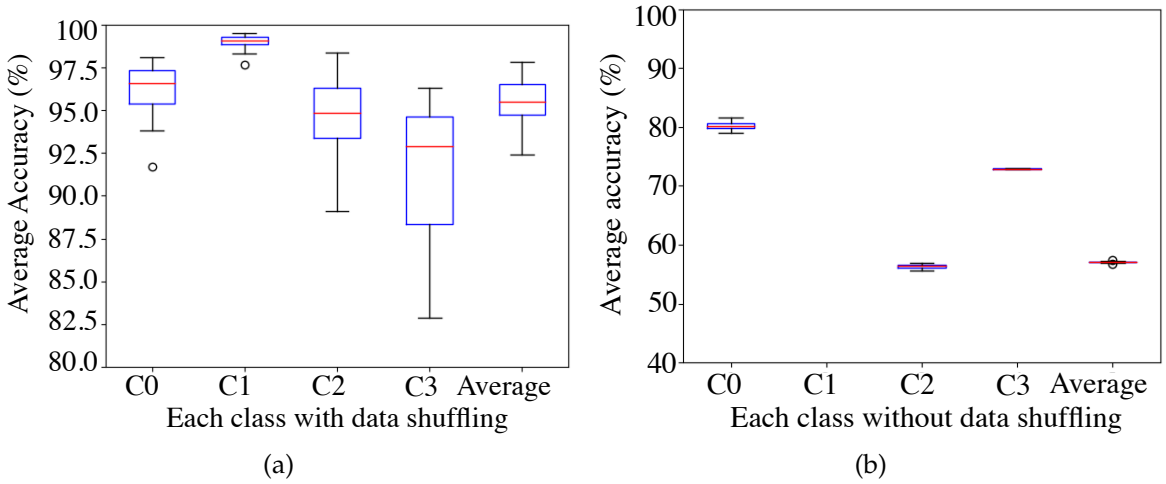


Figure 3.21: The effect of shuffling input data for training a GAN based model: (a) With shuffling and (b) Without shuffling.

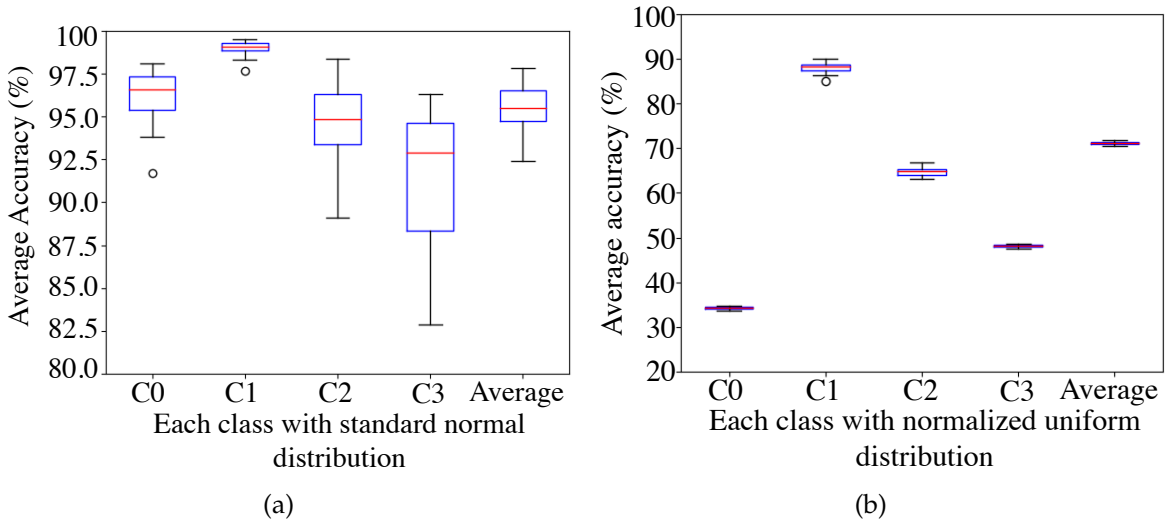


Figure 3.22: The effect of the distribution used for sampling the input z of the GAN generator: (a) standard normal distribution and (b) normalized uniform distribution.

(d) On the distribution used for sampling random inputs

In a GAN-based model, the z signal presented to the generator (recall Fig. 3.13) can be drawn from any distribution. However, for this particular application, some distributions are better than others for the training process. Fig. 3.22 shows the classification results for RF-GAN when a) z is sampled from the standard normal distribution (0 mean and variance 1) and b) an uniform distribution in [-1,1]. No doubt that the former outperforms the latter.

(e) On the initial conditions

GAN is trained using a gradient-based method that is sensitive to initial conditions (weights). Fig 3.23 illustrates the impact of the initial conditions on the fault classification results. As shown in that figure, RF-GAN was able to produce an acceptable accuracy for any of the initial sets of weights used. However and as expected for local optimization methods, in 30 independent runs (initializations), it was possible to identify a particular set of initial random weights that outperformed all the others.

For generating these results, we spent about 80 hours in google colab with the following configurations:

- 1 GPU of Tesla T4.

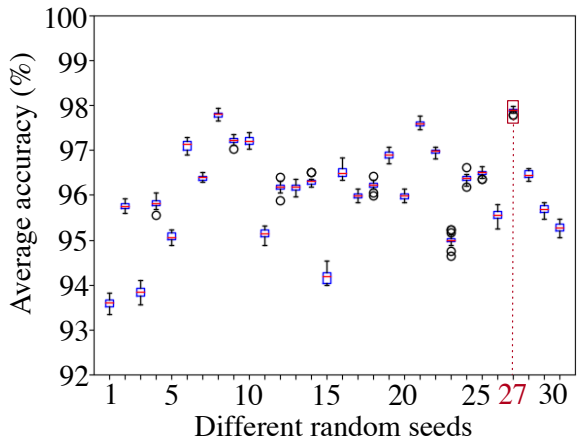


Figure 3.23: The effect of initial weights (generated from different random generator seeds) on the classification accuracy of RF-GAN.

- The tensorflow version of 1.15.1.

3.7 Conclusion

In this chapter, we survey the state of the art of GAN and elaborate on several perspectives on GAN, i.e., theory, variants of GAN, metrics, applications, and open research problems. Due to the adoption of the adversarial learning, GAN has received more and more attention in the AI community. It brings the theory of a two-player min-max game between two agents into the generative model so that the model can generate real-like samples through adversarial learning.

However, GAN is hard to train since three main obstacles exist: mode collapse, training oscillation and vanishing gradient. The possible solutions to these challenges are designing an efficient model by choosing appropriate network architectures, loss functions, and metrics. Even though we state that many different GANs variants have been proposed to avoid the above problem, some issues still remain. As a powerful generative model, GAN does not explicitly estimate the distribution of real samples. The ability to generate new samples from latent variables significantly applies to machine vision, image processing, machine translation, speech recognition, information security, or fault diagnosis.

Moreover, a case study is introduced that applies GAN as an oversampling method

for fault diagnosis. Since robots are wildly used in the industry and their maintenance and monitoring systems are resorting more and more to data intensive machine learning methods. Methods such as MLP, CNN, ESN or DBM have all been used for such endeavours. However, all these methods rely on a representative, balanced, and large enough training set, which due to the very nature of (some) faults is very hard to collect from the equipment. Motivated by the recent success of generative adversarial networks (GANs), in this study, we have applied, for the first time, this type of generative model as an oversampling method for data augmentation in an industrial robot.

A comprehensive empirical analysis was performed, taking into account six different scenarios for mitigating the imbalanced data, including classical under and oversampling (SMOTE) methods. In all of these, a WPT combined with GAN is used for feature generation while an RF is used for fault classification. Studies were also conducted to assess the sensibility of aspects such as generator selection, the number of training examples in each class, training data shuffling, the distribution used for sampling input random data, and initial conditions.

The main conclusion is that it was possible to increase the performance of the fault diagnosis for an industrial robot for any of the GAN-based models over classical undersampling and oversampling (SMOTE) methods. This is accomplished at the expense of a much higher design and computational effort. Training GAN is not an easy task due to the model collapse and other factors and it is certainly a quite time-consuming process. After training a GAN for each fault, one will have a set of generators able to produce synthetic data in an efficient way. Within the GAN-based models, those that keep track of the best current generator during training yielded the best results. No statistically significant difference was observed between the scenario that uses exclusively synthetic data for the faulty states and the scenario that uses the available real data for such states. This is yet another piece of evidence on the quality of the obtained GAN generators.

In many cases like prognostics and health management (PHM), enough data can effectively improve the monitoring capability of the industrial system. But it can not

completely be executed due to a lack of faulty data. GAN is an efficient tool to get rid of the limitation of data imbalance state, which can enhance the monitoring capability in PHM. Therefore, this study provided a data background for PHM. However, this approach still has limitations, one is that this model can only learn the data distribution from a limited faulty data source while there are many kinds of faulty data in the industrial system. For the new faulty data, it needs to be sent to this model to learn the new distribution again to generate enough data that will bring time cost for training. Another one is that this approach is trained by sending one single faulty class as input to GAN to obtain a generator. That means we need to train several GAN models for several faulty classes, which is computationally demanding and time-consuming.

3.8 Appendix

3.8.1 Random forests for fault classification

Ensemble learning uses a group of algorithms to get a better prediction than any of its base algorithms. A random forest (RF) is a homogeneous ensemble classifier that uses a set of decision trees (DT) [168, 169]. Each DT is grown independently using the Bagging technique. In addition, and to increase diversity (reducing the correlation between trees), a RF grows each tree from a random selection of data features. Once trained a RF uses a majority voting mechanism for making its classification (or regression) prediction. The CART algorithm is frequently used to grow a decision tree. In the CART algorithm, the Gini index is the metric used for selecting the data set feature to be used in a given node of the tree. Given a node $\hat{m}$ and the estimated probability $p(c|\hat{m})(c = 1, 2, 3, \ldots, C)$, the Gini impurity index is defined as:

$$G(m) = \sum_{c_1 \neq c_2} p(c_1|\hat{m})p(c_2|\hat{m}) = 1 - \sum_{c=1}^{C} p^2(c|\hat{m}) \tag{3.50}$$

Let $\hat{n}$ be the splitting point of node $\hat{m}$, that separates the node into two portions in which a proportion P_a of the samples in m is assigned to $\hat{m}_a$ and a proportion P_b

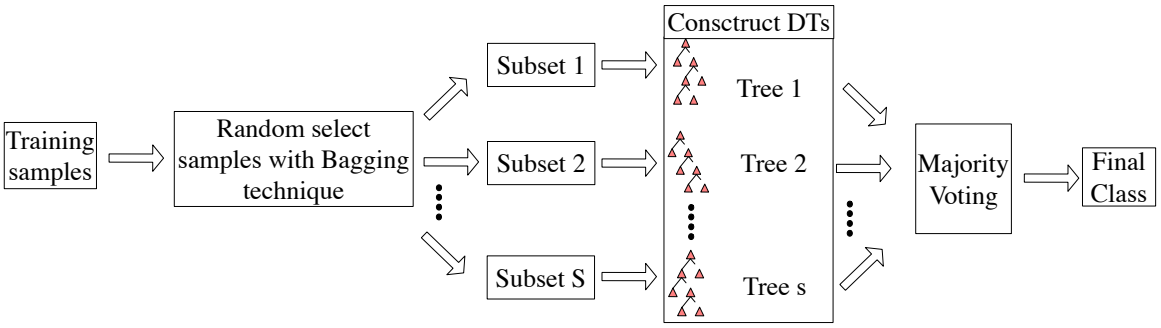


Figure 3.24: Steps for building a random forest

is assigned to $\hat{m}_b$, i.e., $P_a + P_b = 1$. Thus, the decrease in the Gini impurity index is defined as follows:

$$\delta G(\hat{m}, \hat{n}) = G(\hat{m}) - P_a G(\hat{m}_a) - P_b G(\hat{m}_b) \tag{3.51}$$

The optimal feature j^* and the optimal splitting point $\hat{n}^*$ that produces the largest decrease in the Gini impurity corresponds to

$$\hat{n}^*, j^* = \arg\max \delta G(\hat{m}, \hat{n}) \tag{3.52}$$

The flowchart for building an RF is shown in Fig. 3.24.

4

VGAN: a V-matrix based generative adversarial network

In which we introduce a V-matrix based loss function for GAN to handle the training instability, design an early stopping like mechanism for the generator, and present experimental results and conclusions.

4.1 Introduction

As a powerful framework, generative adversarial network (GAN) have the ability to learn non-trivial distributions from random signals and real-world examples [29]. However, one problem with GANs is the training oscillation [170]. This is related to the simultaneous and adversarial training of two neural networks composing the GANs.

For mitigating the oscillation effect, the loss function is a critical issue. The original GAN resort to the KL divergence [171] to measure the distance between the generated and real distributions. To improve the training process, some variants were proposed that concentrated on improving the loss function, such as WGAN [103] that resorts to the EMD or Wasserstein-1 distance, Bi-GAN [172], GAN using MSE loss [40], cGAN [30], WGAN-GP[104], LSGAN [173] and AC-GAN [174].

Another issue in GAN design is the stop criterion for selecting the "best" data generator. The naif selection of the generator obtained in the last epoch might simply not produce the desired results due to the above mentioned oscillation effect. In other words, the desired generator may have appeared in an earlier iteration during the training, not necessarily in the last one.

This chapter addresses both of the mentioned design and training issues in GANs with the following main contributions:

i) For mitigating loss function oscillations during training, motivated by both the theoretical background and the empirical evidence obtained in classification and regression problems, a V-matrix based regularization is used within the cGAN framework. The V-matrix based criterion proposed by Vapnik *et al*.[38, 39] generalizes the well-known and widely used MSE criterion. In the same vein, our proposed GAN framework, VGAN, generalizes both the MSE GAN [40] and the WGAN-GP frameworks (Section 4.2.2);

ii) A novel early stopping like a strategy that keeps track during training of the most suitable model (Section 4.2.3);

iii) The application of the proposed VGAN to an industrial robot fault diagnosis where the VGAN is used as a data augmentation tool to cope with an imbalanced data set (Section 4.4). Results show that VGAN outperforms nine other scenarios including vanilla GAN, conventional regularization and SMOTE, a classic data augmentation technique. Furthermore, the early stopping like mechanism allows to obtain a monotonic increasing performance of the model during training and, when combined with the proposed regularization, yields the highest fault classification accuracy among all other scenarios.

The remainder of this chapter is organized as follows. The proposed framework is presented in section 4.2, including conditional WGAN, VGAN and the novel stop criterion. The experimental settings such as neural network architecture, hyper-parameter settings and considered scenarios are detailed in section 4.3. Results and discussion are analyzed in Section 4.4. Finally, conclusions is addressed in Section 4.5.

4.2 Methodology

4.2.1 On GAN, Wasserstein GAN and conditional Wasserstein GAN

Both the generator G and the discriminator D forming a GAN are simultaneously refined through an adversarial learning process. The goal of the learning process is to allow the generation of synthetic data via G (P_g distributed) that are indistinguishable from real data (P_{data} distributed). The equilibrium $P_{\text{data}} = P_g$ can be achieved with

$$G^\star, D^\star = \arg \min_G \max_D V(G, D) \tag{4.1}$$

where max denotes the maximization of the distribution of the discriminator, min stands for the minimization of the probability in the generator, and $V(G, D)$ is the loss function.

Typically, traditional GAN with KL-based loss function (Eq. (3.7)) can not ensure a

better performance which WGAN (Eq. (3.24)) can improve this strongly oscillates.

The conditional version of GAN, cGAN (refer to subsection 3.3.4), is designed for giving some controls over the generation process [30]. Specifically, cGAN adds extra information, say y, to both the generator G and the discriminator D. The loss function of cGAN can be found in Eq. (3.26).

A vanilla GAN model is learned through unsupervised learning and its generator generates data for a single distribution. If examples are required from k different distributions k GANs are need to be trained; a quite inefficient and time consuming task. A cGAN can simplify this task as it uses only one model for data generation even from different distributions. This is accomplished by resorting to the label information of the real-world examples. By using this information the learning becomes a supervised learning process. Due to this, several improvements can be conceived for the learning process. One possibility is to adopt a WGAN-GP type of loss function (see equation 3.25) in which case (3.26) becomes:

$$\begin{aligned} V_{\text{cWGAN-GP}}(G,D) =& E_{x\sim P_{\text{data}}}[D(x|y)] + E_{z\sim P_G}[1 - D(G(z|y)|y)] + \\ & \lambda_{gp} E_{x\sim P_{\text{data}}}[(||\nabla D(x|y)|| - 1)^2] \end{aligned} \tag{4.2}$$

4.2.2 VGAN

In the mitigation of oscillations during adversarial training, the loss function plays a critical role in GAN. To address this problem, the previous WGAN used Wasserstein distance as the loss function for better data generation. But this measurement still has some drawbacks in generating desired signals that need to be further improved. To address this problem, we are proposing the adoption of a V-matrix based regularization criterion within the conditional WGAN framework that considered the mutual point for data generation. The V- matrix based criterion proposed by Vapnik et al [38, 39] generalizes the well-known and widely used MSE criterion. In the same vein, our proposed GAN framework, VGAN, generalizes both the MSE GAN [13] and the WGAN-GP frameworks. This criterion can be viewed as a generalization of the widely used mean square error criterion. For brevity the idea is presented below for the sim-

plest case. For a more general treatment see [39].

The binary classification problem can be viewed as the problem of estimating the conditional probability of class $y = 1$ given an observation $\boldsymbol{x} \in \mathbb{R}^d$, $p(y = 1|\boldsymbol{x})$. Following [39], we can *rethink* the computation of $f(\boldsymbol{x}) = p(y = 1|\boldsymbol{x})$ by viewing it as a solution of the Fredholm integral equation [175]:

$$\int_{\mathbb{R}^d} \theta(\boldsymbol{x} - \boldsymbol{x}')f(\boldsymbol{x}')\mathrm{d}p(\boldsymbol{x}') = p(y = 1, \boldsymbol{x}) \tag{4.3}$$

where the kernel $\theta(z) = 1$ if $z \geq 0$; 0 otherwise. Both the unknown cumulative distribution functions $p(\boldsymbol{x})$ and $p(y, \boldsymbol{x})$ can be estimated from iid data $\{\boldsymbol{x}_i, y_i\}_1^N$ sampled from $p(y, \boldsymbol{x})$ with $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$. In particular, the empirical estimates can be given respectively by

$$\hat{p}(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N} \theta(\boldsymbol{x} - \boldsymbol{x}_i) \tag{4.4}$$

$$\hat{p}(y = 1, \boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N} y\theta(\boldsymbol{x} - \boldsymbol{x}_i) \tag{4.5}$$

Using these estimates and from (4.3) one has:

$$\sum_{i=1}^{N} \theta(\boldsymbol{x} - \boldsymbol{x}_i)f(\boldsymbol{x}_i) = \sum_{i=1}^{N} y_i\theta(\boldsymbol{x} - \boldsymbol{x}_i) \tag{4.6}$$

Solutions for the above equation can be found by minimizing

$$\rho^2 = \rho(f)^2(\sum_{i=1}^{N} \theta(\boldsymbol{x} - \boldsymbol{x}_i)f(\boldsymbol{x}_i), \sum_{i=1}^{N} y_i\theta(\boldsymbol{x} - \boldsymbol{x}_i)) \tag{4.7}$$

that, when ρ is viewed as the Euclidean distance, translates into

$$\begin{aligned}\rho^2 &= \int (\sum_{i=1}^{N} \theta(\boldsymbol{x}-\boldsymbol{x}_i) f(\boldsymbol{x}_i) - \sum_{i=1}^{N} y_i \theta(\boldsymbol{x}-\boldsymbol{x}_i))^2 \mathrm{d}\boldsymbol{x} \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} f(\boldsymbol{x}_i) f(\boldsymbol{x}_j) V(i,j) + \\ &\quad \sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j V(i,j) - 2\sum_{i=1}^{N}\sum_{j=1}^{N} f(\boldsymbol{x}_i) y_j V(i,j)\end{aligned} \tag{4.8}$$

where V = $[V(i,j)]$ is the so-called V-matrix and

$$V(i,j) = \int \theta(\boldsymbol{x}-\boldsymbol{x}_i)\theta(\boldsymbol{x}-\boldsymbol{x}_j)\mathrm{d}\boldsymbol{x} \tag{4.9}$$

In the d-dimensional case where $\boldsymbol{x} \in [0, \boldsymbol{c}]^d$, one has

$$V(i,j) = \int \prod_{k=1}^{d} \theta(\boldsymbol{x}-\boldsymbol{x}_i)\theta(\boldsymbol{x}-\boldsymbol{x}_j)\mathrm{d}\boldsymbol{x} \tag{4.10}$$

Furthermore, (4.10) with $\boldsymbol{c} = (c_1, \ldots, c_d)^T$ can be represented as

$$V(i,j) = \prod_{k=1}^{d} (c_k - \max(x_i^k, x_j^k)) \tag{4.11}$$

where c_k represents the maximum value of the k-th coordinate ($k = 1, \ldots, d$) while x_i^k denotes the k-th coordinate of the i-th example ($i = 1, \ldots, n$). However, this multiplicative form is hard for high-dimensional space [39]. In this work, the alternative additive form of (4.11) is used, i.e.,

$$V(i,j) = \sum_{k=1}^{d} (c_k - \max(x_i^k, x_j^k)) \tag{4.12}$$

From (4.8) we have

$$\rho_{\text{V}}^2 = \sum_{i=1}^{N}\sum_{j=1}^{N}(y_i - f(\boldsymbol{x}_i))(y_j - f(\boldsymbol{x}_j))V(i,j) \tag{4.13}$$

Notice that (4.13) holds as a special case the classical mean square error

$$\rho_{\text{MSE}}^2 = \sum_{i=1}^{N}(y_i - f(\boldsymbol{x}_i))^2 \tag{4.14}$$

whenever the V-matrix equals the identity matrix. That is, while in (4.14) only the residuals $\Delta_i = y_i - f(\boldsymbol{x}_i)$ are taken into account while searching for f, in (4.13) both the residuals and the relative dispersion of observations $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are considered, resulting in a more general criterion.

Now, aiming at smoothing the learning process, motivated by the above development, and by the empirical evidence from some classification and regression problems (e.g., [38, 39]), we regularize (4.2) using the (4.13) as regularizer yielding

$$V_{\text{VGAN}}(G,D) = V_{\text{cWGAN-GP}}(G,D) + \gamma\rho_{\text{V}}^2(G) \tag{4.15}$$

where γ is a user-defined regularizing coefficient. Notice that the regularizer $\rho_{\text{V}}^2(G)$ concerns only the generator G, where relatively to (4.13) $\boldsymbol{x}_i$ and y_i are now the generator i-th input and corresponding desired output, respectively. For brevity, hereafter (4.15) is referred to as V-regularization, and this type of generative model is VGAN. Notice that VGAN can be viewed as a generalization of the MSE GAN [40] and degenerates into it for $\gamma = 1$ and V-matrix equal to the identity matrix. Also, VGAN can be viewed as a generalization of the cWGAN-GP (4.2) and degenerates into it for $\gamma = 0$. To the best of our knowledge this is the first study where the V-matrix is applied in the realm of GANs and deep learning in general. For comparison purposes, in Section 4.4, the

traditional MSE based regularization, i.e., MSE GAN

$$V_{\text{MSE}}(G, D) = V_{\text{cWGAN-GP}}(G, D) + \gamma \rho^2_{\text{MSE}}(G) \tag{4.16}$$

is also considered.

4.2.3 On the early stopping in GANs

Classification accuracy is a metric that summarizes the performance of a classification model as the proportion of correct predictions divided by the total number of predictions. Inspired by this, we proposed new stop criteria named early stopping, which used the idea of classification accuracy for evaluating the performance of GAN. Roughly speaking, early stopping means interrupting the learning process whenever a theoretical, such as a bias-variance trade-off, or, e.g., a holdout validation-based criterion is verified.

We follow a holdout strategy where the available real data is divided into disjoint training and validation sets. In the following, these are denoted as $\{\mathbf{x}^{(t)}, \mathbf{y}^{(t)}\}$ and $\{\mathbf{x}^{(v)}, \mathbf{y}^{(v)}\}$, respectively, the superscripts $^{(t)}$ and $^{(v)}$ referring to the training and validation sets, respectively. The input data of the training set is $\mathbf{x}^{(t)} = \{x_1^{(t)}, x_2^{(t)}, \ldots, x_N^{(t)}\}$ while the label information is given by $\mathbf{y}^{(t)} = \{y_1^{(t)}, y_2^{(t)}, \ldots, y_N^{(t)}\}$. A similar notation is used for the validation set.

Let K be the number of epochs. Thus, K successive generators will be available during learning. The generator available at iteration j, G_j, $(j = 1, 2, \ldots, K)$ generates $\mathbf{x}_j^{(g)} = \{x_{j1}^{(g)}, \ldots, x_{jN}^{(g)}\}$ when $\mathbf{y}^{(t)}$ is presented at its input. Thus, we can write

$$\mathbf{x}_j^{(g)} = G_j(z|\mathbf{y}^{(t)}) = G_j(\mathbf{y}^{(t)}) \tag{4.17}$$

Notice that the superscript $^{(g)}$ refers to *generated* data; as before, z is a random variable (e.g., Gaussian).

Now consider a classifier, say a random forest R, the formulation of R can be shown

as

$$M = \{\hat{n}^*, j^* | I(\hat{m}, \hat{n})\} \tag{4.18}$$

where I stands for the Gini index, m is the tree node, $\hat{n}$ refers to the splitting node, $\hat{n}^*$ represents the optimal splitting node and j^* is the optimal feature.

For each j of the K generated data sets $\mathbf{x}_j^{(g)}$, train a model M_j, s.t.,

$$M_j = R(\mathbf{x}_j^{(g)}) = R(G_j(\mathbf{y}^{(t)})) \tag{4.19}$$

At this time, employ the validation set to evaluate the j-th model as follow:

$$y_j^{(p)} = M_j(\mathbf{x}_j^{(v)}) \tag{4.20}$$

where the superscript $^{(p)}$ refers to a predicted label.

The performance of the j-th model M_j is viewed as a proxy for the performance of the j-th generator G_j. We use accuracy (Acc) for evaluating such performance. The Acc of the j-th model is computed in a straightforward way:

$$Acc(G_j) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}(y_{ji}^{(t)}, y_{ji}^{(p)}) \tag{4.21}$$

where the $\mathbb{1}(\alpha, \beta)$ is the indicator function that returns 1 if α equals β; 0 otherwise.

We term $G_{\max}$ the generator whose generated data $\mathbf{x}_j^{(g)}$, yields the max accuracy, i.e.,

$$G_{\max} = \arg \max_{G_j; j=1,\ldots,K} Acc(G_j) \tag{4.22}$$

At this point, one can legitimately ask why using all of this if one can directly use the GAN discriminator to compute the accuracy. The reason is twofold. On one hand, during learning (especially in the initial epochs) the discriminator is not yet properly trained, so it is still a poor option for evaluating generated data; On the other hand, if we would get a poor performance we were not able to tell which of the generator or discriminator would be to blame. So by using a classifier such as random forest (See

Appendix 3.8) for evaluating the generated data we can best control the selection of $G_{\max}$. Notice that the discriminator is still and indeed an important part of the GAN learning process as only simultaneous learning of the generator and the discriminator allows for the best results [176].

4.3 Experiments

4.3.1 The MNIST data set

The MNIST shown in Fig. is a digit handwriting data set of 60,000 training and 10,000 testing images [177]. It contains ten digits from 0 to 9. Each image has a size of 28×28 pixels. We randomly extract 10,000 images from the data set for out-of-sample validation. In the experiments, 6000 examples are used for training while other 4000 are used for out-of-sample validation.

(a) Neural network architectures

The neural network architectures of the generator and the discriminator are shown in Fig. 4.1. As shown in Fig. 4.1, in the generator, the input random distribution is firstly combined with label information and then send to one Dense layer (also called fully connected layer) and reshape to the 2-D feature map. Finally, it will send to three CNN blocks (CNN block up) to generate synthetic images. In the discriminator, the input image (real or synthetic one) will embed with label information and then send to 2 CNN blocks (CNN block down) and finally flattened to get the outputs. Moreover, the details of CNN block down and CNN block up are given in 4.1 (c).

(b) Hyper-parameter settings

The learning rate for both the generator and the discriminator was set to 2×10^{-4}. The Adam optimizer was used with key parameters $\beta_1 = 0.5$, $\beta_2 = 0.9999$ in both the discriminator and the generator. A maximum number of $K = 10,000$ iterations were considered.

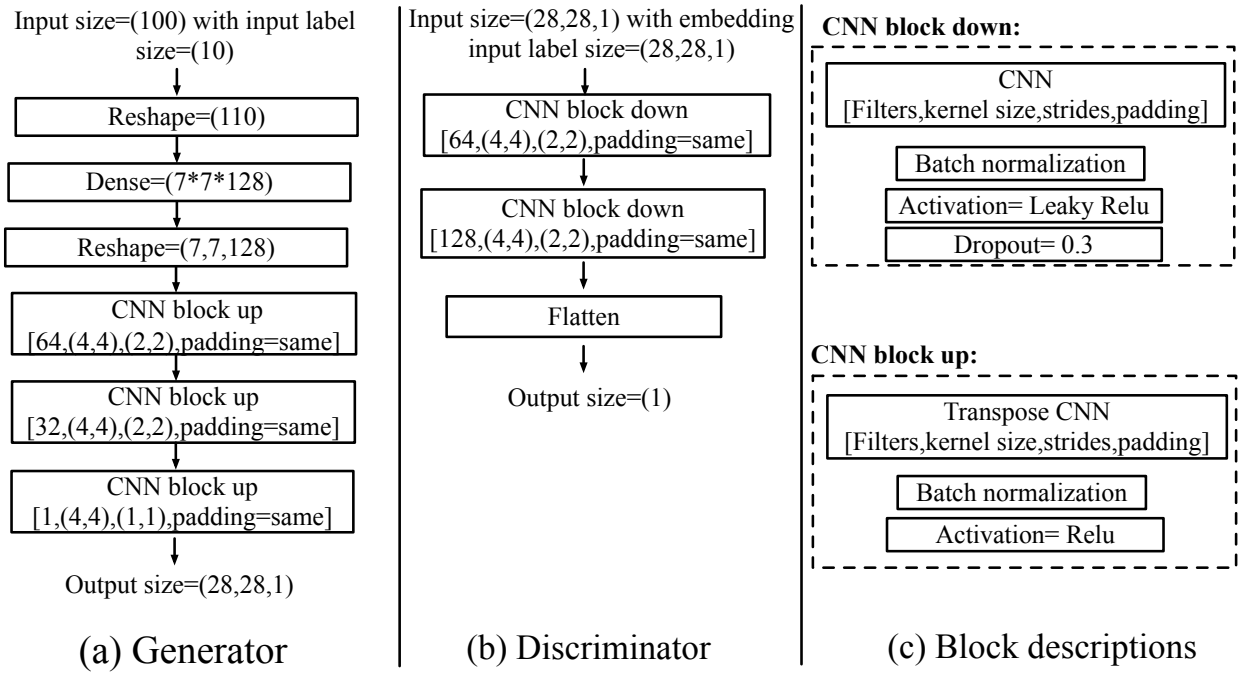


Figure 4.1: The architecture of VGAN for MNIST dataset

4.3.2 The industrial robot data set

In the previous chapter, we used GAN as a data augmentation tool to deal with the fault diagnosis of the industrial robot. We know that this activity is critically dependent on the existence of a balanced data set which is obliviously hard to be obtained in a real world. To validate the performance of VGAN, we also start from a perspective of the fault diagnosis under the imbalanced data set (see Table. 2.2) with the same faults configuration (C_0,C_1,C_2,and C_3). In the training phase, the training data is used together with fault label information in the training of the VGAN. After training the VGAN, $G_{\max}$ (4.22) is used for generating enough synthetic data for achieving an augmented balanced training set. This is the data set used for training the fault classifier. The validation data set which was never used during training is subsequently used for assessing the fault classification performance.

(a) Neural network architectures

In this application, both the discriminator and the generator are multi-layer-perceptrons with a fully connected topology. With the label information, the topology of the gener-

ator is 67:1024:128 while that of the discriminator is 131:1024:2048:1.

(b) Hyper-parameter settings

The learning rate for the generator is set to 2×10^{-4}, while for the discriminator is set to 1×10^{-4}. These were selected empirically after some preliminary tests. The Adam optimizer was used with key parameters $\beta_1 = 0.9$, $\beta_2 = 0.9999$ in both the discriminator and the generator. A maximum number of $K = 20,000$ iterations were considered. The V-matrix regularization coefficient γ is discussed in Section 4.4.4 below.

4.3.3 Considered scenarios

Several scenarios were taken into consideration to assess the effectiveness of the proposed model. In all these scenarios, all the hyper-parameter of each GAN are the same as mentioned before except for the loss function. Besides, we use the additive form (4.12) of V-matrix in VGAN. We used an SVM (support vector machine) classifier with an RBF (radial basis function) kernel for fault classification. For early stopping, we use a random forest for selecting (See 3.6.2) $G_{\max}$ (4.22). The different scenarios are:

i) *normal*, i.e., cWGAN-GP loss (4.2) and early stopping, i.e., the employed generator given by (4.22);

ii) *normal_l*, i.e., cWGAN-GP without early stopping, i.e., the employed generator corresponding to the generator obtained in the last iteration;

iii) *mse*, i.e., MSE GAN (4.16) and early stopping;

iv) *mse_l*, i.e., MSE GAN without early stopping;

v) *v*, i.e., VGAN (4.15) and early stopping;

vi) *v_l* , i.e., VGAN without early stopping

4.4 Results and discussion

In this section, results on the above mentioned issues such as the imbalanced data set problem, the convergence problem, and the sensitivity of the algorithm to relevant hyper-parameters are presented. The subsequent discussion takes into account confusion matrices and statistical tests of hypotheses.

4.4.1 Comparisons of the different scenarios

To draw statistically significant conclusions on the results obtained from the different scenarios, the Friedman test was applied and the the Wilcoxon test is used as the *posthoc* (see subsection 3.6.4 (a)).

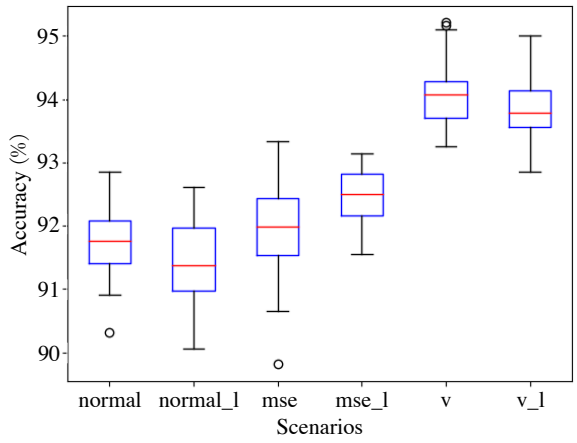


Figure 4.2: Boxplots exhibiting the distribution of accuracy over 30 independent runs for the different scenarios.

As the Friedman test for the results presented in Fig. 4.2 yielded $p_{\text{Friedman}} = 8.79 \times 10^{-61} < 0.05$, we must conclude that a statistically significant difference exists between the six scenarios. Consequently, we present in Table 5.1 the obtained Wilcoxon *post hoc* results.

From Table 4.1 one can see that scenario v (VGAN with early stopping) has 5 wins meaning that it ranks first among all the others.

From Fig. 4.2 we observe that median accuracy is i) normal: 91.756%; ii) normal_l: 91.483%; iii) mse: with 91.961%; iv) mse_l: 92.47%; v) v: 92.93%; and vi) v_l: 92.36%. Also, with or without early stopping, VGAN outperforms all the other scenarios. More

Table 4.1
Wilcoxon *post-hoc* results for the six studied scenarios.

Comparison	**p-value**	**Winner**
normal vs. normali_l	0.032	normal
normal vs. mse	0.0198	-
normal vs. mse_l	6.88×10^{-5}	mse_l
normal vs. v	1.7×10^{-6}	v
normal vs. v_l	1.73×10^{-6}	v_l
normal vs. mse	0.02	mse
normal vs. mse_l	9.77×10^{-6}	mse_l
normal vs. v	1.73×10^{-6}	v
normal vs. v_l	1.73×10^{-6}	v_l
mse vs. mse_l	0.01	mse_l
mse vs. v	1.73×10^{-6}	v
mse vs. v_l	1.73×10^{-6}	v_l
mse_l vs. v	1.73×10^{-6}	v
mse_l vs. v_l	$1.7 \times 10^{-6}3$	v_l
v vs. v_l	0.0125	v

specifically, scenario v) has an accuracy 1.169% higher than the normal scenario i).

The confusion matrices for the different scenarios are presented in Fig. 4.3. As observed in Fig. 4.3, all these matrices concern the 6000 validation set mentioned in Table. 2.2. In each matrix the diagonal refers to the correct classification rate in the corresponding class. All the other entries refer to mis-classifications. From these figures one can see also that confusion matrix for scenario v (VGAN with early stopping) has the best classification performance.

To further illustrate the performance, t-Distributed Stochastic Neighbour Embedding (t-SNE) is used to characterize the fault data generation that is represented in Fig. 4.4. As shown in Fig. 4.4, the green part is C_1, the blue part is C_2 and the rest part is C_3. From Fig. 4.4, Both of the clustering performance is worse except 4.4 (e), which is our proposal in this work. This is an evidence that our data generation method is more appropriate for fault data generation than any other model.

Besides, the calculation burden of the above scenarios is presented in Table. 4.2, it

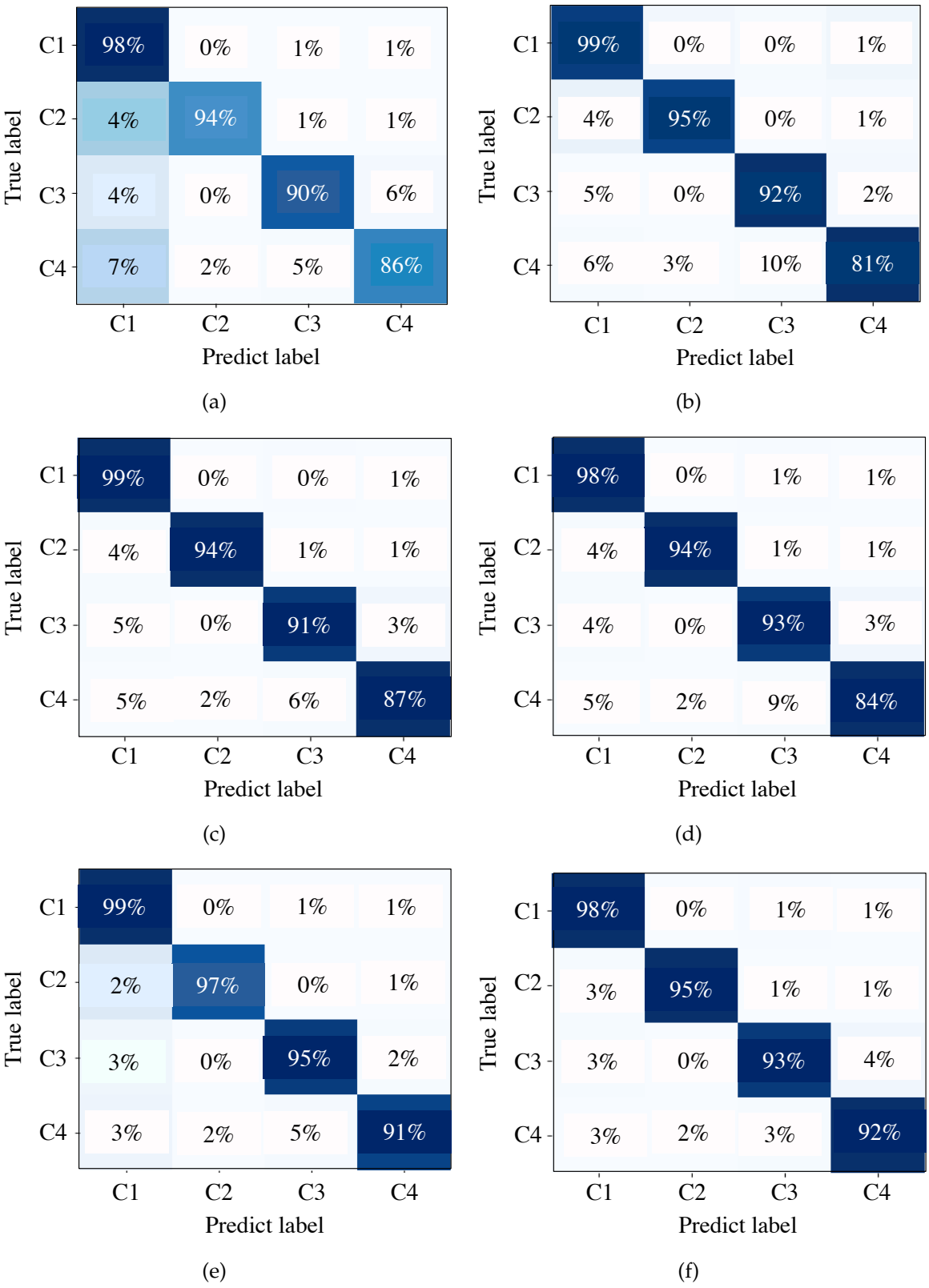


Figure 4.3: Confusion matrices for scenario (a) Normal; (b) Normal_l; (c) mse; (d) mse_l; (e) v; and (f) v_l.

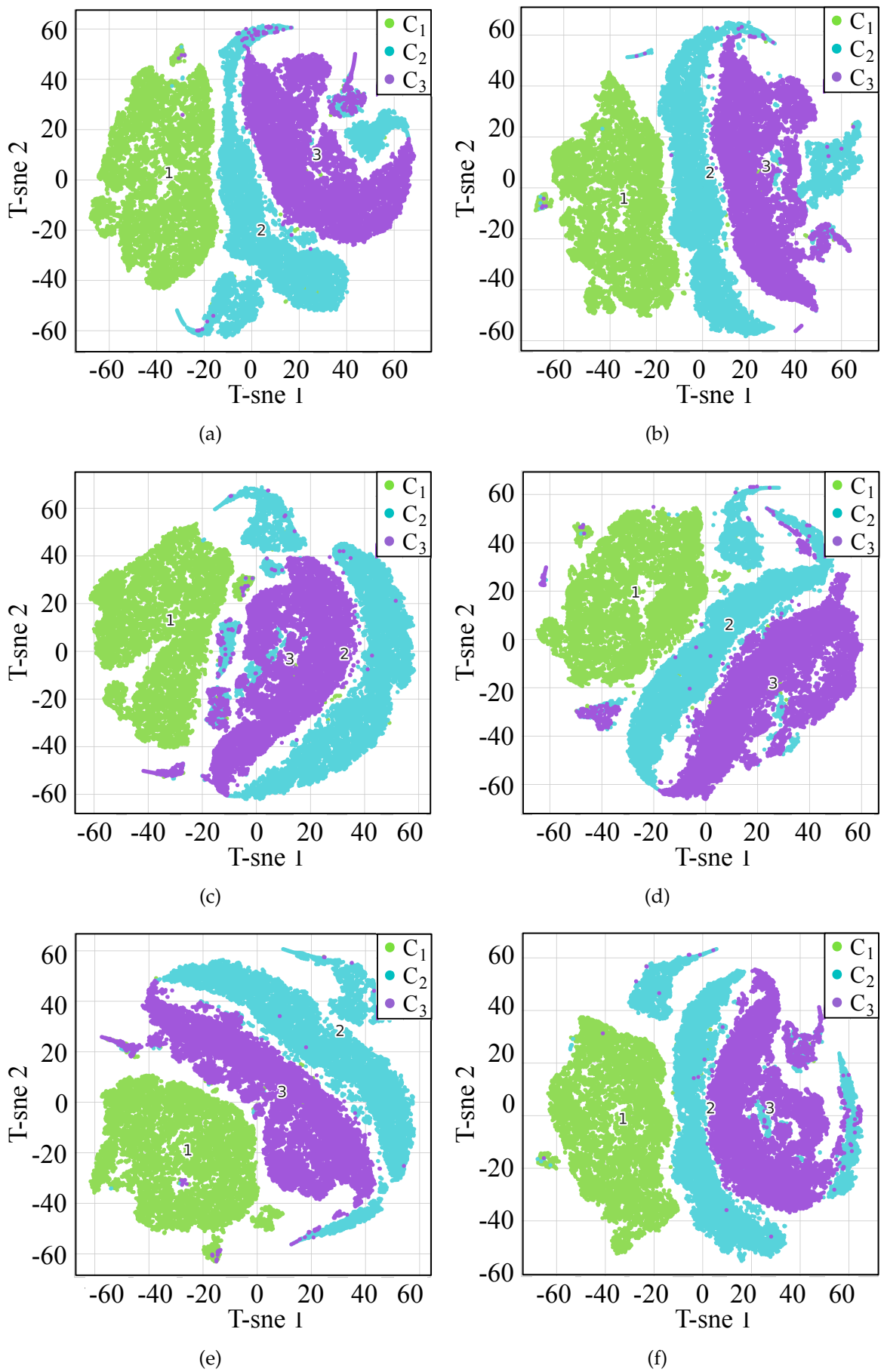
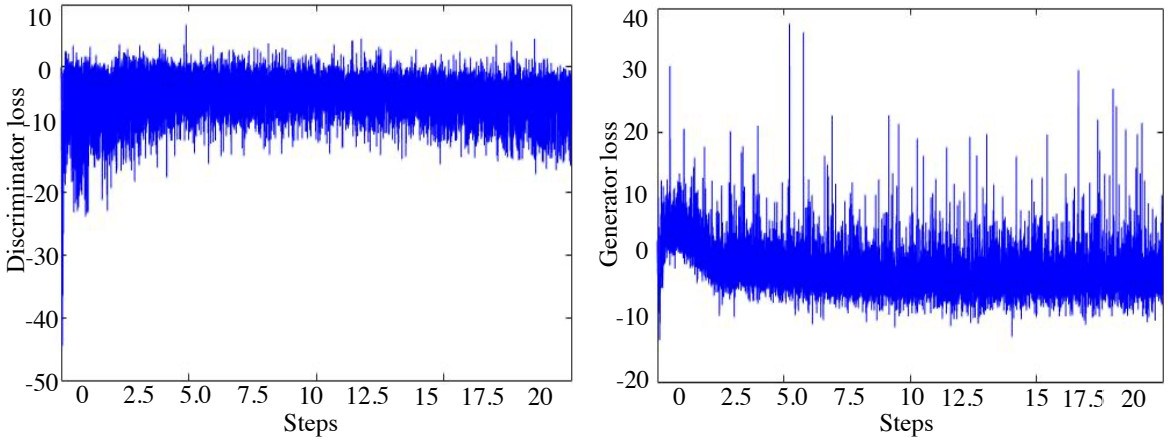


Figure 4.4: t-SNE representation of each data generation scenario: (a) normal; (b) normal_l; (c) mse; (d) mse_l; (e) v and (f) v_l

Table 4.2
Calculation burden via different scenario for fault data generation

Scenario	normal	norma_l	mse	mse_l	**v**	v_l
Times (seconds)	2905	3152	2463	3092	**2138**	3653



(a) (b)

Figure 4.5: Typical convergence curves for VGAN with early stopping: (a) discriminator and (b) generator.

can be found that with our early stopping and the V-matrix based loss function, this model can quickly find the desired generator for optimal data generation.

4.4.2 Convergence curves

Fig. 4.5 shows typical convergence curves for scenario v (VGAN with early stopping). As shown in Fig. 4.5 the loss curves of both discriminator and generator are still oscillating. We hypothesize that this is an inevitable side effect of adversarial learning between the generator and the discriminator. Notice however that, without V-regularization, oscillations are way worse: Fig. 4.6 shows for the non-regularized case that while the loss evolution looks similar to Fig. 4.5 the loss ranges are much smaller ([-35; 5] *vs* [-50; 10] for the discriminator; [-15; 10] *vs* [-20; 40] for the generator).

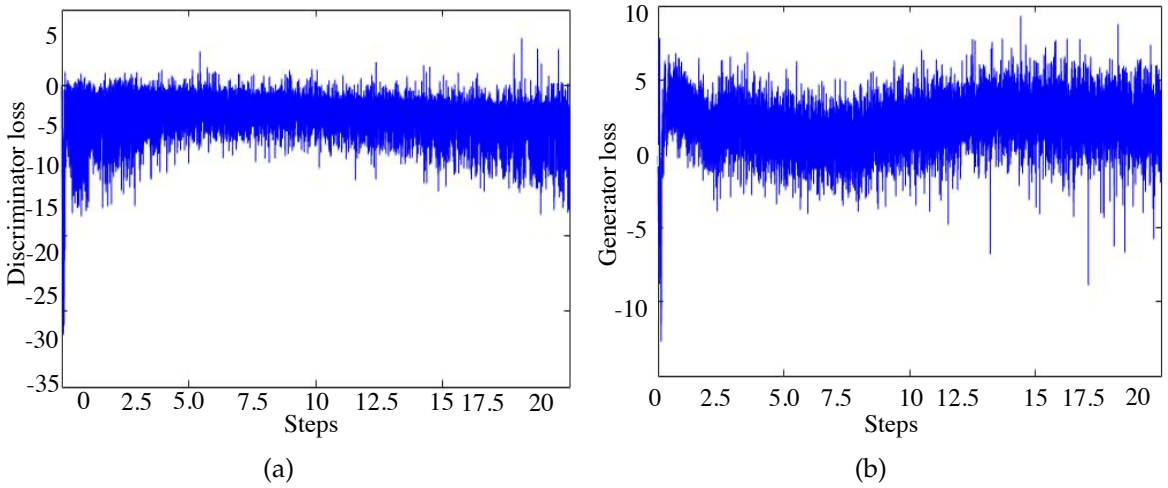


Figure 4.6: Typical convergence curves for cWGAN-GP without early stopping: (a) discriminator and (b) generator.

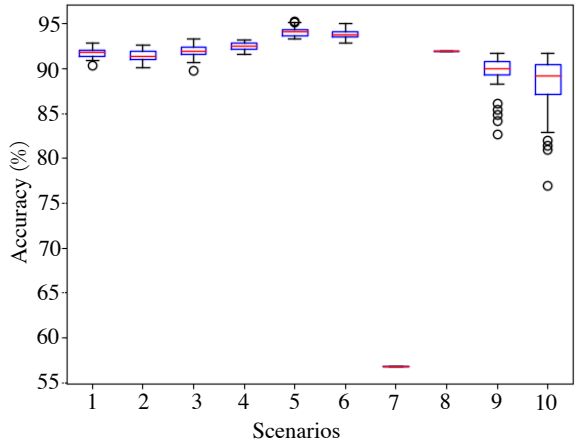


Figure 4.7: Results of the difference scenarios for dealing with the imbalance training data; see text for details.

4.4.3 On imbalance data sets

To assess the effectiveness of the model when dealing with an imbalanced data set, ten scenarios were studied, as follows: 1: normal; 2: normal_l; 3: mse; 4: mse_l; 5: v; 6: v_l (all the above with the previous presented meaning); 7: imbalance training set, i.e., 14000 health examples and only 1% of examples (140) for each fault; 8: imbalance training set, as above, but processed by the SMOTE technique; 9: vanilla unsupervised GAN with early stopping as per [131]; 10: vanilla unsupervised GAN without early stopping; In all these scenarios, an SVM classifier with the RBF kernel was used for fault classification.

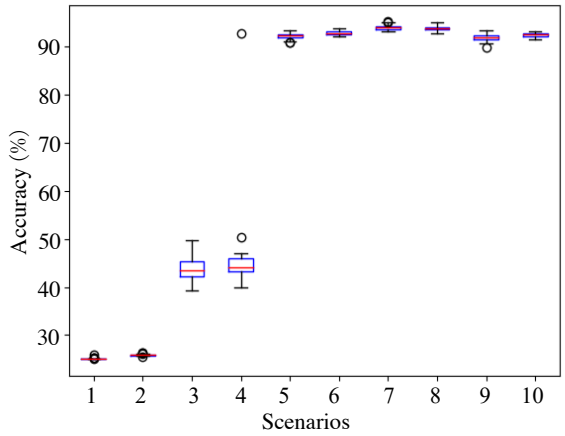


Figure 4.8: Accuracy values for the v model (VGAN with early stopping) and v_l model (VGAN without early stopping) for different values of γ in (4.15).

.

From the above Fig. 4.7, we can see that the 5th model (VGAN with early stopping) has the highest median accuracy. The Friedman test reveals a statistical significant difference $p_{\text{Friedman}} = 0.012 < \alpha = 0.05$.

4.4.4 On the model sensitivity

Below we present a sensitivity discussion on four relevant aspects of the model: regularization term, ratios between healthy and faulty data, weight initialization, and accuracy vs. epochs.

Regularization term: To evaluate the effect of the V-regularization coefficient γ in (4.15), we compared a v model (VGAN with early stopping) with v_l (VGAN without early stopping). Both scenarios share the same hyper-parameter settings except for γ that takes values in $\{1\,e^{-1}, 1\,e^{-2}, 1\,e^{-3}, 1\,e^{-4}, 0\}$ see Fig. 4.8. In this figure, 1 to 10 stands for ve1, ve1_l, (), ve2, ve2_l, ve3, ve3_l, ve4, ve4_l, 0, 0_l, respectively. Besides, Labels ve1, ve1_l correspond to $\gamma = 1\,e^{-1}$, ..., ve4, ve4_l to $\gamma = 1\,e^{-4}$, while labels 0 and 0_l correspond to $\gamma = 0$ in a v and v_l model, respectively.

The value of $\gamma = 1\,e^{-4}$ yielded the highest accuracy values among all other cases, including the case where $\gamma = 0$ corresponds to the cWGAN-GP model (4.2). Another observation is that early stopping is a better option.

Friedman and Wilcoxon *post hoc* tests are also performed for evaluating the effect

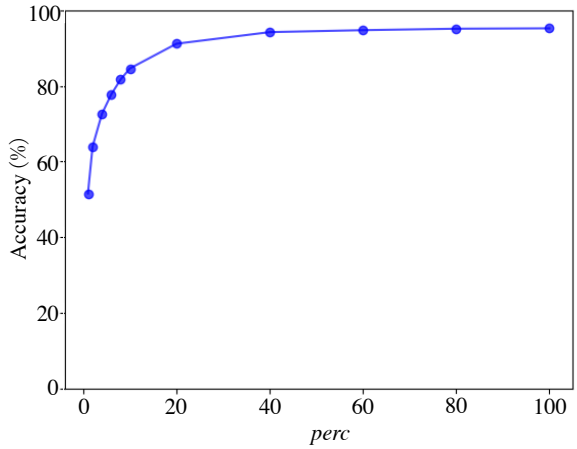


Figure 4.9: A learning curve taking into account a percentage *perc* of the faulty examples relatively to the number of healthy examples.

of the regularization term. The ten scenarios in Fig. 4.8 yield $p_{\text{Friedman}} = 6.21 \times 10^{-61}$ meaning that there exists a significant difference among them. Afterward, the *post hoc* analysis using Wilcoxon test shows that the regularization term $\gamma = 1\,e^{-4}$ with early stopping (ve4 model) with 9 wins ranks first among all the others.

Faulty vs. healthy data ratios: To illustrate the importance of using a balanced data set in this application, we evaluate the performance of the SVM fault classifier under different faulty to healthy data ratios. As always, the number of healthy examples in the training set is 14000 and the initial number of examples in each fault class is 140 as per Table. 2.2. The following percentages were considered $perc = 1, 2, 4, 6, 8, 10, 20, 40, 60,$ 80, and 100%. For instance, when *perc*=10%, only $14000 \times 10\% = 1400$ examples were used for each fault class. From these 1400-140=1260 examples were generated with our best model (VGAN with early stopping). The resulting learning curve is shown in Fig. 4.9. The average accuracy increased from 56.25% to 95.17% by increasing the availability of faulty data from 1% (140 examples) to 100% (14,000 examples). That is, the imbalanced data set had a strong negative impact on the fault classifier. Furthermore, there was a 20%+ increase in performance in the range $1\% \leq perc \leq 40\%$; afterward the improvement in accuracy was slower and slower. For $perc \geq 80\%$ the improvement was neglectable. That is, adding more data after that point hardly improved the performance.

Weight initialization: In our experiments we considered 30 different random seeds

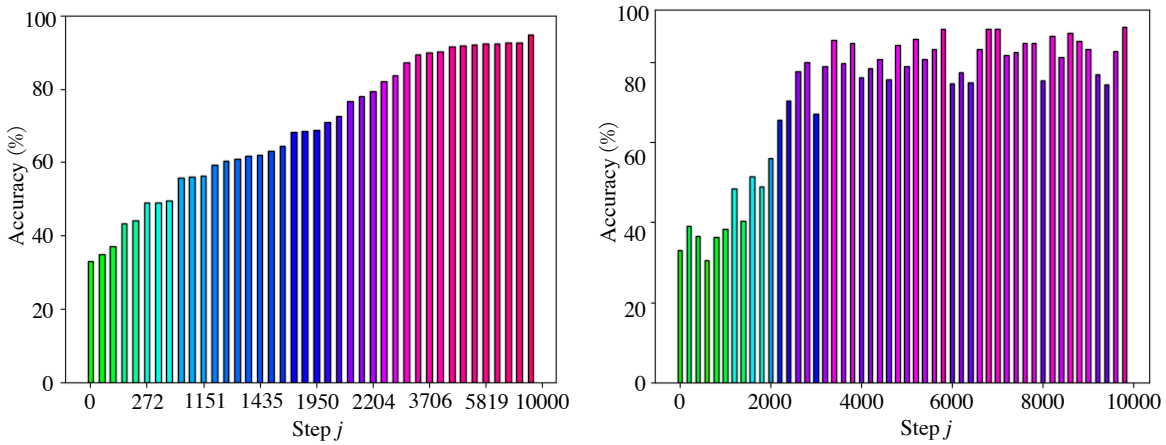


Figure 4.10: Classification accuracy obtained for VGAN using (4.22) as generator, as a function of the epochs.

for weight initialization. The impact of the different initial weights are reflected in the dispersion of accuracy as presented in the boxplots of the above figures.

Accuracy vs. epochs: Fig. 4.10)(a) shows the classification accuracy obtained using (4.22) as a generator as a function of the current number of epochs for VGAN. From Fig. 4.10 as the number of epochs increased a monotonic increase in the accuracy is observed, from 38% up to 93.28%.

When the model v_l (VGAN without early stopping) is used in Fig. 4.10(b), we can see that the accuracy in each epoch is not monotonic increasing.

4.4.5 Results with MNIST data set

With the same scenarios described in 4.3.3, the results of the boxplot using the MNIST data set are shown in Fig. 4.11.

From Fig. 4.11 we observe that median accuracy is i) normal: 72.53%; ii) normal_l: 61.50%; iii) mse: with 72.56%; iv) mse_l: 58.54%; v) v: 77.50% ; and vi) v_l: 66.91%. Also, with or without early stopping, VGAN outperforms all the other scenarios. More specifically, scenario v) has an accuracy 4.97% higher than the normal scenario i).

For generating the above results, we spent around 210 hours in google colab with the following configurations:

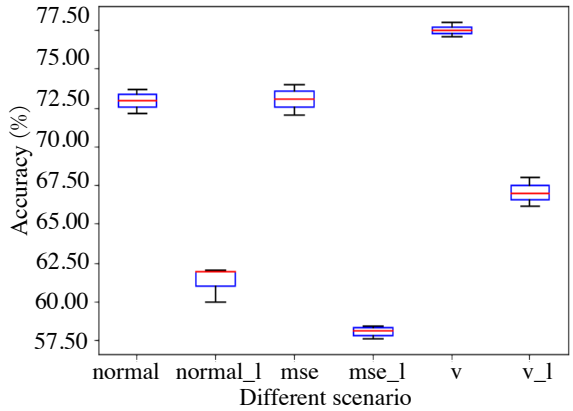


Figure 4.11: Boxplots exhibiting the distribution of accuracy over 3 independent runs for the different scenarios.

- 1 GPU of Tesla V100.
- The tensorflow version of 1.15.1.

4.5 Conclusion

Generative adversarial networks have a great potential for applications ranging from image processing to machinery fault diagnosis. However, in general, it is still hard to train them to obtain a generator with the desired performance. With this in mind, we proposed VGAN, a V-regularized conditional Wasserstein GAN with gradient penalization. V-regularization is based on a criterion that generalizes the mean square error criterion in the sense that it takes into account not only residuals but also the dispersion among the independent variable in input data. VGAN generalizes both MSE GAN and cWGAN-GP. A novel early stopping like strategy was also proposed.

The effectiveness of the proposed formalisms was illustrated through a comprehensive set of experiments on a fault diagnosis task for an industrial robot where VGAN was used as a data augmentation tool for dealing with imbalanced data sets. Results show that the VGAN outperforms nine other scenarios including vanilla GAN, conditional WGAN-GP with and without MSE based regularization, and SMOTE, a classic data augmentation technique. Furthermore, the early stopping like mechanism allows one to obtain a monotonic increasing performance of the model during training and,

when combined with the V-regularization, ranks first in terms of classification accuracy among all other scenarios.

4.6 Appendix

This appendix specifies the used procedures. In the light of reproducible research, the python code used for generating the presented MNIST examples is publicly available from https://github.com/pzq522362451/VGAN

5

SW-CylcyeGAN: a sliced Wasserstein distance-based cycle consistency generative adversarial network

In which we explore an alternative loss function for cycleGAN named the sliced Wasserstein distance to replace Wasserstein distance for fast convergence, design a new metric for qualitative evaluation of GAN, present experimental results and main conclusions.

5.1 Introduction

In its vanilla form, a GAN is formed of two units, a generator and a discriminator. The former generates synthetic data given a random distribution while the latter tries to distinguish between real and the generated synthetic data. Both modules are trained simultaneously, aiming at achieving an equilibrium where the discriminator cannot differentiate between real and synthetic examples meaning that the generator has learned the real data distribution.

The loss function is recognized as a crucial factor in the efficiency of GANs training [36]. Both the losses of the generator and the discriminator oscillate during adversarial learning. In addition, when the discriminator gets more accurate than the generator, it may be possible that the whole system does not learn as gradients become unavailable for updating the weights. Designing an appropriate loss function can help GANs to obtain faster and stabler convergence.

Many works have paid attention to the loss function of GAN. The least squares GAN [173] was designed to deal with the vanishing gradient problem. This adopts the least squares loss rather than the divergence-based loss of the conventional GAN. Geometric GANs was proposed in [178] based on the idea of the support vector machine to search for a decision boundary between real and synthetic examples with the hinge loss [179, 180]. Zhu *et al*. [33] proposed a cycle consistency GAN (CycleGAN, hereafter also referred to as unconditional CycleGAN) that combines two traditional GANs for the unpaired image-to-image transference and that avoids mode collapse. Che *et al*. [181] introduced several methods of regularizing the loss function, which can improve the training of GAN.

Another type of approach is found in Wasserstein GAN (WGAN) [103] as it resorts to the Wasserstein-1 distance (WD for short), also called EMD [182]. In [104], the gradient penalty was introduced to improve the performance of WGANs. More recently, Pu *et al*. [41] proposed VGAN that uses a regularization based on Vapnik V-matrix generalizing WGAN with gradient penalty.

The sliced Wasserstein distance (SWD) is based on random projections [183] to

quantity the two distributions difference with a low-level computational cost compared to WD. SWD has previously been applied in vanilla GAN [184] and in auto-encoders [185]. Motivated by the obtained results in those works and by the need to avoid mode collapse, in this chapter, we investigate the usage of SWD in CycleGANs. This distance is first applied to unconditional CycleGANs and then extended to conditional CycleGANs. These models are evaluated with both a public data set (MNIST) and in the data set of the industrial robot.

The main contributions of the chapter are:

1: The sliced Wasserstein distance is applied, for the first time, in the development of unconditional and conditional CycleGANs aiming at smoother, faster, more efficient convergence while addressing mode collapse;

2: To the best of our knowledge, it is the first time that either unconditional or conditional CycleGAN (either with Wasserstein or sliced Wasserstein distance) is used to transfer healthy states to different fault states for addressing the imbalanced data problem in fault diagnosis of an industrial robot.

The remaining of this chapter is organized as follows. section 5.2 details the sliced Wasserstein CycleGAN. section 5.3 introduces the experimental settings, including nenural network architectures, hyper-parameter settings and the considered scenarios. section 5.4 interprets the results and discussions. Finally, section 5.5 draws some conclusions.

5.2 Methodology

The designed SW-CycleGAN is presented in subsection 5.2.1. The metric for our proposal is presented in Section 5.2.1 (d). The whole procedure of the proposal is illustrated in Section 5.2.2

5.2.1 Sliced Wasserstein CycleGAN

In this section, we first present in sequence, the sliced Wasserstein distance, a SWD-based unconditional CycleGAN (SW-CycleGAN) and its conditional variant. The mechanisms of the residual networks are also presented at the end.

(a) The sliced Wasserstein distance

The Wasserstein-p distance between distributions P_d and $G_\theta(P_z)$ is given by [186]:

$$W_p(P_d, G_\theta(P_z)) = \inf_{\gamma \in \Pi(P_d, G_\theta(P_z))} (\mathop{\mathbb{E}}_{(x,y)\sim\gamma}[||x-y||^p])^{\frac{1}{p}}, \tag{5.1}$$

where for p is a user-defined positive integer, x denotes real data points, y represents synthetic one from $G_\theta(P_z)$, $\Pi(P_d, G_\theta(P_z))$ stands for the set of all joint distributions $\gamma(x,y)$. Informally, $\gamma(x,y)$ denotes how much 'mass' must be transposed from x to y in order to transform the distribution P_d into $G_\theta(P_z))$. With Kantorovich-Rubinstein duality, (5.1) yields

$$\begin{aligned} W_p(P_d, G_\theta(P_z)) = &\sup_{||f_w||_L <= 1} \mathop{\mathbb{E}}_{x\sim P_d}[f_w(x)] \\ &- \mathop{\mathbb{E}}_{z\sim P_z}[f_w(G_\theta(z))], \end{aligned} \tag{5.2}$$

where the sup is defined over all 1-Lipschitz functions $f_w : \chi \to \mathbb{R}$. Therefore, (5.2) can be used in (3.24) in the definition of WGAN.

With this in mind, consider the case $p \neq 1$ in (5.1). Suppose that data x belongs to the real data set $\mathcal{D}$ and the generated data $\hat{x} = G_\theta(z)$ belongs to a synthetic data set $\mathcal{F}$ ($\mathcal{D}$ and $\mathcal{F} \in$). The minimum distance estimation is defined as [187]:

$$\arg\min_\theta |P_d, G_\theta(P_z)|, \tag{5.3}$$

where $|P_d, G_\theta(P_z)|$ denotes a divergence between probability distributions. Using a Wasserstein measurement, (5.3) can be re-written as

$$\arg\min_\theta W_p, \tag{5.4}$$

Considering the Wasserstein quadratic distance $W_2^2(\mathcal{D}, \mathcal{F})$ between two distributions, (5.1) can be re-written as [188]

$$W_2^2(\mathcal{D}, \mathcal{F}) = \frac{1}{|\mathcal{F}|} \min_{s \in \sum_{|\mathcal{F}|}} \sum_{i=1}^{|\mathcal{F}|} ||x_{s_{\mathcal{D}}(i)} - \hat{x}_{s_{\mathcal{F}}(i)}||_2^2, \tag{5.5}$$

where $\sum_{|\mathcal{F}|}$ is the set of all projections of $|\mathcal{F}|$ points, $s(i)$ is the index needs to be searched for mapping $\hat{x}$ to x. For facilitating the convergence of (5.5), the optimal searching for s^* is defined as an integer linear program with constant matrices M, i.e.,

$$W_2^2(\mathcal{D}, \mathcal{F}) = \frac{1}{|\mathcal{F}|} \min_{M} \sum_{i=1}^{|\mathcal{F}|} \sum_{j=1}^{|\mathcal{D}|} M_{i,j} ||x_{s_{\mathcal{D}}(j)} - \hat{x}_{s_{\mathcal{F}}(i)}||_2^2, \tag{5.6}$$

where the matrix M is also known as a permutation matrix that can be estimated through a linear programming solver. For approximation between these two distributions with permutation matrix M, a summation over random directions of vectors Ω in the projection sphere (Fig. 5.1) is subject to the following sorting conditions [184]:

$$x^r_{s_D(i)} \leq x^r_{s_{\mathcal{D}}(i+1)}, \exists i \in \{0 \leq i < |\mathcal{D}|\}, \tag{5.7}$$

$$\hat{x}^r_{s_{\mathcal{F}}(i)} \leq \hat{x}^r_{s_{\mathcal{F}}(i+1)}, \exists i \in \{0 \leq i < |\mathcal{F}|\}, \tag{5.8}$$

As illustrated in Fig. 5.1, the data and synthetic points are projected into the one-dimensional spaces by sorting the projections of all possible directions r_i on the projection sphere Ω [188]:

$$SW_2^2(\mathcal{D}, \mathcal{F}) = \oint_{r_i \in \Omega} W_2^2(\mathcal{D}^{r_i}, \mathcal{F}^{r_i}) dr, \tag{5.9}$$

Therefore, the approximation for sliced Wasserstein distance SWD_2 with all possible directions over Ω can be found by computing the average distance between sorted samples [189]:

$$SWD_2 = \min_{\theta} \frac{1}{|\Omega|} \sum_{r_i \in \Omega} W_2^2(\mathcal{D}^{r_i}, \mathcal{F}^{r_i}), \tag{5.10}$$

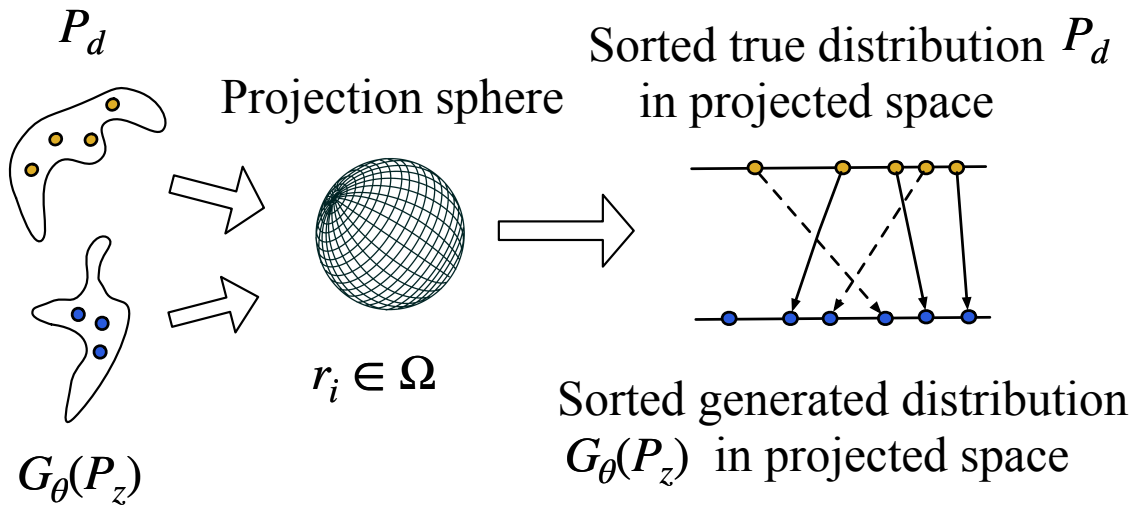


Figure 5.1: Random projections and permutation of two distributions P_d and P_z, adapted from [3]

.

Specifically, combined with (5.5), the above expression can be re-written as

$$SWD_2 = \min_\theta \frac{1}{|\Omega|} \sum_{i=1}^{|\mathcal{F}|} ||x^r_{s_\mathcal{D}(i)} - \hat{x}^r_{s_\mathcal{F}(i)}||_2^2. \tag{5.11}$$

(b) Unconditional cycle generative adversarial networks

In [33], a new variant of GAN, CycleGAN, was proposed for unpaired image-to-image translation[190]. A CycleGAN is composed of two classic GANs. Since these two GANs have a symmetric architecture, only one part of the CycleGAN is detailed in Fig. 5.2. For illustrative purposes, consider the MNIST dataset. In the first GAN, real images (say images of '3') from domain A are sent to the generator G12 to produce synthetic images from domain B (say images of '0'), which in turn are compared with real domain B images in the discriminator D12. At the same time, the generated domain B images will be passed through another generator G21 to reconstruct images of the domain A. The loss between real domain A images and reconstructed domain A images is the cycle consistency loss 12. The second GAN uses real images from domain B to generate images for domain A via generator G21. Adversarial training will compare the generated domain A images with real domain A images through the discriminator D21.

The involved losses of a CycleGAN are [33] :

$$V_{CycleGAN} = L_{adv}^{12} + L_{adv}^{21} + \lambda_c * L_{cyc}, \tag{5.12}$$

where L_{adv}^{12} and L_{adv}^{21} are adversarial losses, L_{cyc} is the cycle consistent loss, λ_c is the corresponding hyper-parameter. The expressions for these losses are as follows.

$$\begin{aligned} L_{adv}^{12} = & \mathop{\mathbb{E}}_{x\sim P_B}[\mathrm{D}_{12}(x_B)] + \mathop{\mathbb{E}}_{x\sim P_A}[(\mathrm{D}_{12}(\mathrm{G}_{12}(x_A))] \\ & + \lambda \mathop{\mathbb{E}}_{x\sim P_A}[(|\nabla \mathrm{D}_{12}(\alpha_{12}x - (1-\alpha_{12}\mathrm{G}_{12}(z)))| - 1)^2, \end{aligned} \tag{5.13}$$

$$\begin{aligned} L_{adv}^{21} = & \mathop{\mathbb{E}}_{x\sim P_A}[\mathrm{D}_{21}(x_A)] + \mathop{\mathbb{E}}_{x\sim P_B}[(\mathrm{D}_{21}(\mathrm{G}_{21}(x_B))] \\ & + \lambda \mathop{\mathbb{E}}_{x\sim P_B}[(|\nabla \mathrm{D}_{21}(\alpha_{21}x - (1-\alpha_{21}\mathrm{G}_{21}(z)))| - 1)^2, \end{aligned} \tag{5.14}$$

$$\begin{aligned} L_{cyc} = & \mathop{\mathbb{E}}_{x_A}[||x_A - \mathrm{G}_{21}(\mathrm{G}_{12}(x_A))||_2] \\ & + \mathop{\mathbb{E}}_{x_B}[||x_B - \mathrm{G}_{12}(\mathrm{G}_{21}(x_B))||_2]. \end{aligned} \tag{5.15}$$

This scheme is particularly effective for dealing with mode collapse as it forces each generator to produce a new output for each new input. However, training for multi-domains (classes) is quite time-consuming as it needs to be repeated for each new domain pairs. In the MNIST example, for each and every digit in domain B.

(c) Conditional variant of cycle generative adversarial network

A conditional variant of CycleGAN can be considered that adds extra information (a label) to each input which is used to specify the desired output [191]. Returning to the MNIST example as illustrated in Fig. 5.2, the first GAN receives a domain A image together with a label $yB \in \{1, 2, ..., 9\}$ at the generator G12 to generate the corresponding yB domain image. Generated domain B images will compare with real domain B images in the discriminator D12. At the same time, generated domain B images and labels yA will be sent to the other generator G21 to reconstruct domain A images.

In the conditional case, the losses are:

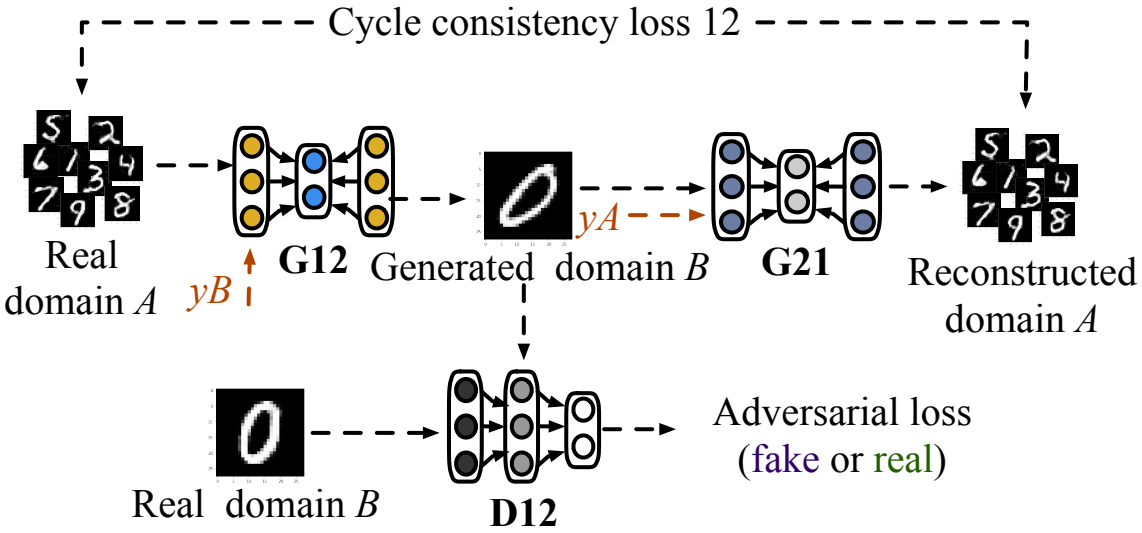


Figure 5.2: The flow chart of a conditional CycleGAN.

$$
\begin{aligned}
L_{adv}^{12} = & \mathop{\mathbb{E}}_{x\sim P_B}[\mathrm{D}_{12}(x_B|yB)] \\
& + \mathop{\mathbb{E}}_{x\sim P_A}[(\mathrm{D}_{12}(\mathrm{G}_{12}(x_A|yB)|yB)] \\
& + \lambda \mathop{\mathbb{E}}_{x\sim P_A}[(\nabla \mathrm{D}_{12}(\alpha_{12}x - (1-\alpha_{12}\mathrm{G}_{12}(x_A|yB))) \\
& - 1|yB)^2,
\end{aligned}
\tag{5.16}
$$

$$
\begin{aligned}
L_{adv}^{21} = & \mathop{\mathbb{E}}_{x\sim P_B}[\mathrm{D}_{21}(x_A|yA)] \\
& + \mathop{\mathbb{E}}_{x\sim P_A}[(\mathrm{D}_{21}(\mathrm{G}_{21}(x_B|yA)|yA)] \\
& + \lambda \mathop{\mathbb{E}}_{x\sim P_A}[(\nabla \mathrm{D}_{21}(\alpha_{21}x - (1-\alpha_{21}\mathrm{G}_{21}(x_B|yA))) \\
& - 1|yA)^2,
\end{aligned}
\tag{5.17}
$$

$$
\begin{aligned}
L_{cyc} = & \mathop{\mathbb{E}}_{x_A}[||x_A - \mathrm{G}_{21}(\mathrm{G}_{12}(x_A|yB)|yA)||_2] \\
& + \mathop{\mathbb{E}}_{x_B}[||x_B - \mathrm{G}_{12}(\mathrm{G}_{21}(x_B|yA)|yB)||_2].
\end{aligned}
\tag{5.18}
$$

(d) Residual networks

To address gradient degradation in deep neural networks, He *et al*. [192] proposed the residual network (ResNet). The ResNet consists of a series of residual blocks, whose

basic diagram is shown in Fig. 5.3 (a), the corresponding expression being

$$x'_{ResNet} = x + Q(x), \tag{5.19}$$

where $Q(x)$ is the residual mapping, x is the input, and x'_{ResNet} is the output from

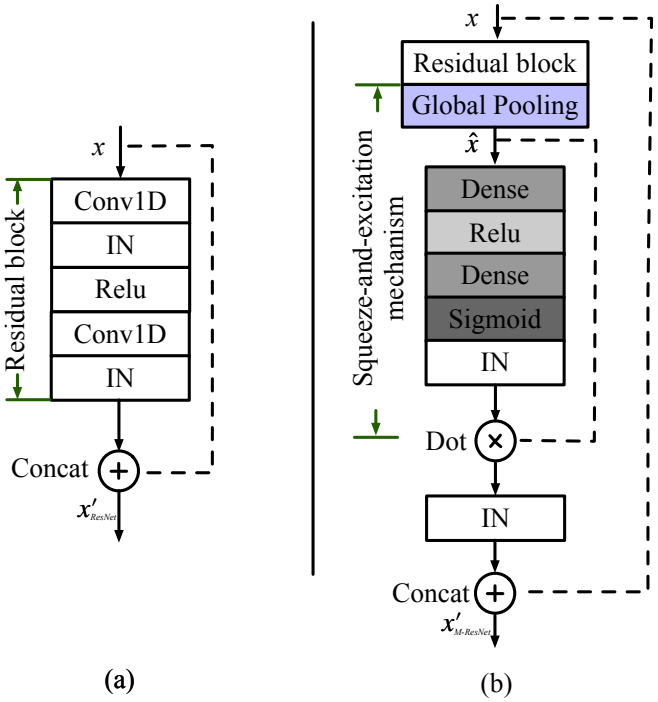


Figure 5.3: The ResNet architecture: (a) the classical ResNet block; (b) the modified ResNet block.

the residual block. Even though this process can release the pressure when adding more layers to the neural network, it may cause a certain insensitivity to the inputs. To this end, a modified ResNet network, i.e., ResNet with the squeeze-and-excitation mechanism (SEM) [193], can be considered to enhance the sensitivity to the inputs. SEM consists of the squeeze block and the excitation block. Firstly, the input vectors are $Q(x)$ and then input to the global average pooling (the squeeze block) is used to shrink inputs in the squeeze stage:

$$\hat{x} = F_{sq}(Q(x)) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} Q(x)(i,j), \tag{5.20}$$

where F_{sq} is the squeeze operation, $\widehat{x}$ is the output from the squeeze block, also known as the input for the excitation block. Then in the excitation block, two dense layers are used to learn the non-linear relationship between each channel, which is shown as

$$s = F_e(\widehat{x}, \mathbf{W}) = f_2(W_2 \times f_1(W_1 \times \widehat{x})), \tag{5.21}$$

where F_e is the excitation operation, f_1 and f_2 are Relu and Sigmoid activations, respectively; W_1 and W_2 are weights for these two dense layers; s is the weight for the feature channel. Next, the output of excitation block s and the $\widehat{x}$ are combined through the so-called 'dot' operation, i.e.,

$$F_{sem} = F_{scale}(\widehat{x} \times s), \tag{5.22}$$

Finally, with instance normalization of F_{sem}, one gets (see also Fig. 5.3 (b)):

$$x^{'}_{M-ResNet} = F_{scale}(F_{sem} + x). \tag{5.23}$$

(e) Metrics for generator selection

Selecting a generator during the adversarial learning is difficult since the convergence curve is not decreasing monotonically, and the generator obtained in the last generator may not be the best one. Therefore, keeping the current best generator is a crucial aspect during GAN training.

The following model compatibility (MC) rate metric can be used to access the generator qualitatively:

$$\begin{aligned} MC_i(\mathcal{D}, \mathcal{F}) &= 1 - \frac{e(\mathcal{D}, \mathcal{F})_{i=K}, K \geq 0}{e(\mathcal{D}, \mathcal{F})_{i=0}}, \\ \textbf{subject to} \quad & e_K < e_0 \end{aligned} \tag{5.24}$$

where i refers to the iteration number, e is the metric measuring how close $\mathcal{D}$, the real

data set, is from $\mathcal{F}$, the corresponding generated data set. We defined e as follows:

$$e_i = \frac{1}{|D|}\sum_{j=1}^{|D|}||x_B(j), G_{12}^i(x_A(j))||_2 + ||x_A(j), G_{21}^i(x_B(j))||_2. \tag{5.25}$$

where $||.||_2$ as the Euclidean metric. The lower the value of this expression the better, meaning that the generated domain has more overlapped with the real domain.

As for the MC rate, initially, e_0 is large but will be reduced with training. The closer the two data sets $\mathcal{D}$ and $\mathcal{F}$ are, the closer e_i will be to zero, the closer $MC_i(D, \mathcal{F})$ will be of to 1. The current best generator is the one with the highest MC.

5.2.2 Procedure of our proposed approach

The whole procedure of the proposed model is shown in Algorithm 1. Besides, the Algorithm for calculating SWD is specified in Algorithm 2.

Algorithm 1 Sliced Wasserstein conditional CycleGAN

input : Mini batch images x_A and y_B in source domain; Mini batch images x_B and y_A in target domain; number of iterations S;batch size K.

output: G_{best}: The generator yielding max MC rate over the considered number of epochs, K.

Set $MaxMC$ rate= 0 **for** $i = 1$ **to** S **do**
  for $j = 1$ **to** K **do**
    Calculate adversarial losses (L_{adv}^{12} and L_{adv}^{21}) by (5.16) and (5.17) with Sliced Wasserstein distance. (**See Algorithm 2**)
    Calculate cycle consistency losses L_cyc with (5.18).
    Obtain the overall losses for conditional CycleGAN through (5.12).
    Calculate the initial L_2 distance with (5.24).
    Update and optimize weights $\theta \in \{\theta_{g12}, \theta_{g21}, \theta_{d12}, \theta_{d21}\}$
    Calculate the updated L_2 distance with (5.24)
    Calculate the current MC rate using with (5.25) **if** $MC > MaxMC$ **then**
      $MaxMC = MC$ $G_{best} = G_i$
    end
  end
end
return G_{best}

Algorithm 2 Sliced Wasserstein distance

input : Images $x \in \{x_A, x_B\}$; Generated images $y \in \{G_{12}(x_A), G_{21}(x_B)\}$; sample size n;number of random projections m, learning rate α.
output: Optimized weight θ_g
while θ_g *not converged* **do**
  Sample random projection directions $\Omega = \{r_{1:m}\}$; Inirial loss L_g=0.
  for *each* $r_i \in \Omega$ **do**
    $x^r \leftarrow \{r^T x_i\}_{i=1}^n$,$y^r \leftarrow \{r^T y_i\}_{i=1}^n$;
    $x_s^r \leftarrow$ *sorted* x^r , $y_s^r \leftarrow$ *sorted* y^r with (5.7) and (5.8) ;
    $L_g \leftarrow L_g + \frac{1}{n}||x_s^r - y_s^r||^2$;
  end
  return $\frac{L_g}{m}$
  $\theta \leftarrow \theta - \alpha \nabla_\theta L_g$
end

5.3 Experiments

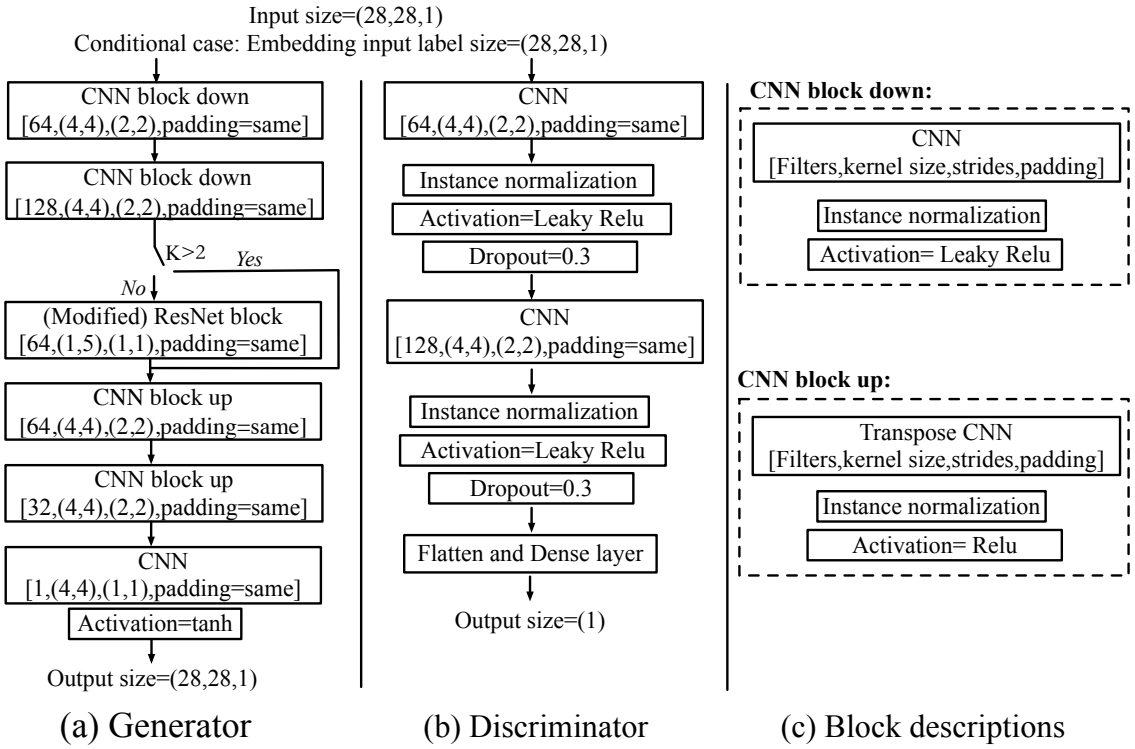
Two data sets are considered: the MNIST data set and an in-house six-of-freedom industrial robot data set acquired for fault diagnosis.

5.3.1 MNIST data set

(a) Neural network architectures

The network architectures of the generator and the discriminator in both CycleGAN and conditional CycleGAN are shown in Fig. 5.4.

As shown in Fig. 5.4, in the unconditional generator, the input images went through two CNN blocks (CNN block down) and two modified ResNet blocks to enhance features. Finally, these features are sent to two transposed CNN blocks (CNN block up) and one transposed CNN for generating synthetic images. The activation of the last layer in the generator is the tanh. Moreover, the details of CNN and the modified CNN blocks are given in 5.4 (c). For the architecture of the unconditional discriminator, the input is the same as the generator as it also receives digit images. The input goes through two convolutional layers with both instance normalization and leaky Relu. The output is flattened through one dense layer to get logits whereas dropout rate was set to 0.3. The architectures for the conditional CycleGAN are similar to the



(a) Generator (b) Discriminator (c) Block descriptions

Figure 5.4: The architecture of CycleGAN for MNIST dataset.

unconditional one just described except for the inputs that were extended with labels.

(b) Hyper-parameter settings

In the experiment of CycleGAN and conditional CycleGAN, the learning rates of both the discriminator and the generator are set to 2e-4 with the Adam optimizer. A sliced Wasserstein distance with 32 random projections ($r = 32$) was considered for the generator loss. The L_2 norm is used in cycle consistency loss with the $\lambda|_c$ set to 10. The batch size is 32, and the number of the max iterations is 1,000 and 10,000 for the unconditional and conditional CyleGAN, respectively.

5.3.2 The industrial robot data set

In this experiment, six different fault conditions were taken into consideration. These conditions are (1) Health condition (C_0); (2) Cracking in Planetary gear A (C_4); (3) Broken tooth in Planetary gear A (C_5); (4) Broken tooth in Sun gear A (C_6); (5) Cracking in Planetary gear B (C_7) and (6) Broken tooth in Planetary gear B (C_8) (See Table 2.1).

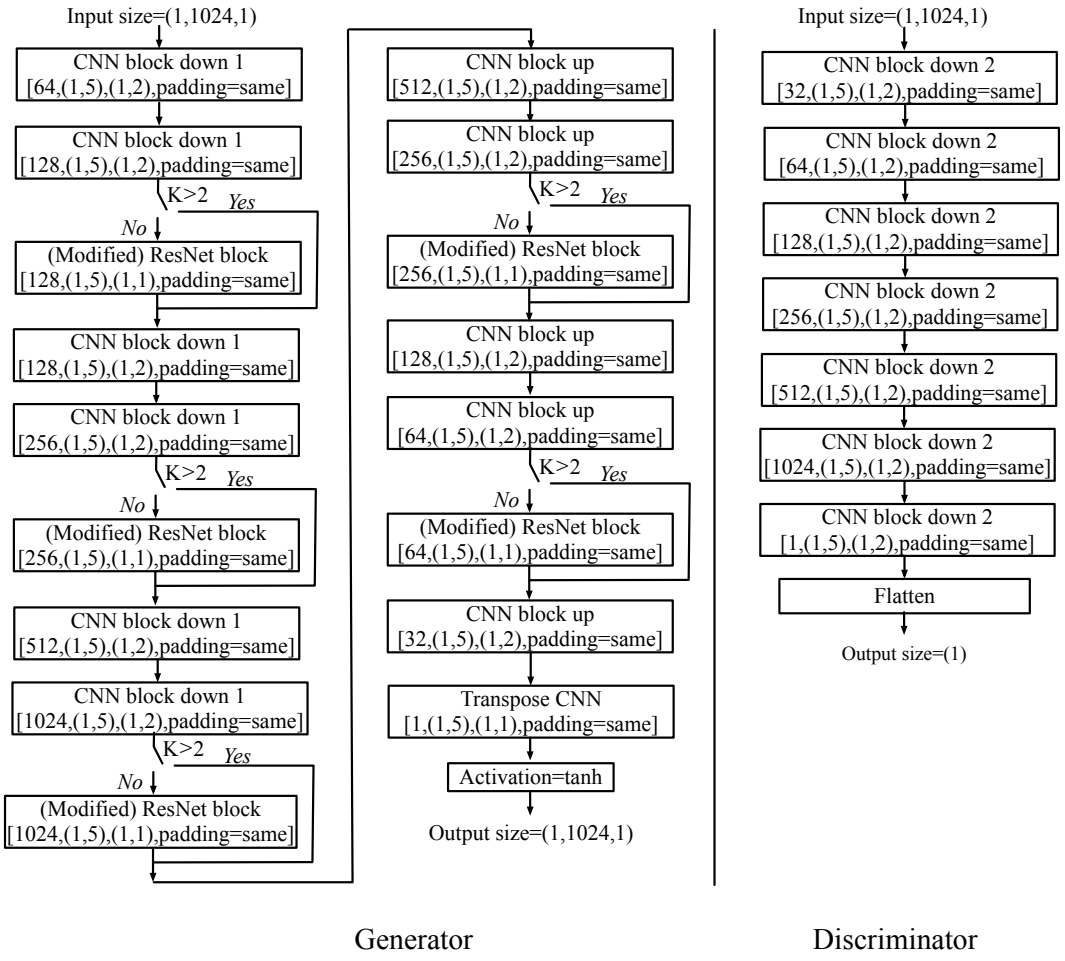
(a) Neural network architectures

The neural network architectures of the generator and the discriminator in both CycleGAN and conditional CycleGAN are shown in Fig. 5.5.

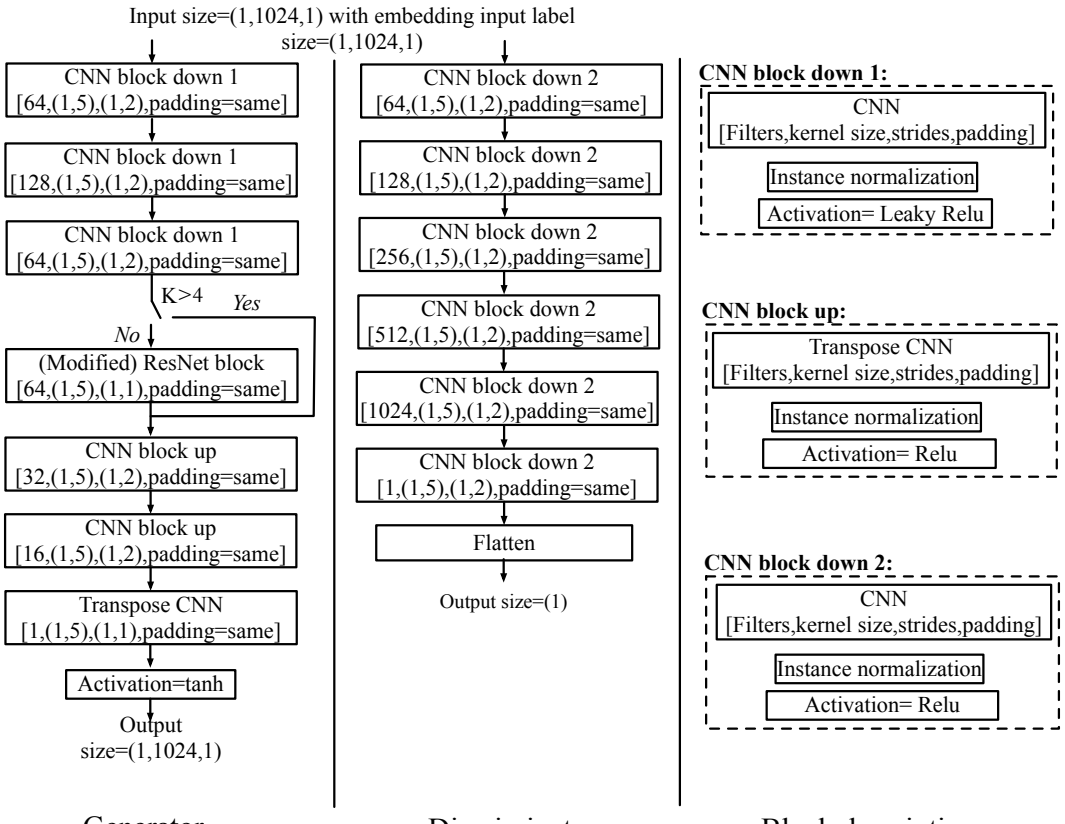
Unlike the MNIST dataset that uses 2-dimensional CNN for computation, the robot data set resorts to 1-dimensional CNN (1D-CNN). For instance, if the input vector has N features, it needs to be reshaped to the shape of (N,1) to be suitable for the 1D-CNN. Therefore, the input for the generator and the discriminator is a series of reshaped vectors. Similar to the above description of the MNIST dataset, the unconditional CycleGAN only receives data from each domain. The features are sent to two CNN blocks and went through two Modified ResNet blocks. Repeat this operation three times to get the output of these blocks. Then the output is sent to two CNN blocks and went through two Modified ResNet blocks (repeat three times) to generate synthetic data. The tanh activation is used at the end of the model. (see Fig. 5.5 (a) Generator). For the discriminator, the input will go through seven CNN blocks (CNN block down 2) and then flattened to get logits (Fig. 5.5 (a) Discriminator).

As for the conditional CycleGAN, the features are extended with labels to sent to three CNN blocks (CNN block down 1) and went through four Modified ResNet blocks. In turn, the output of these blocks is sent to two other transposed CNN blocks (CNN block up) to produce synthetic data. Again, in the last layer, the tanh activation is applied (see Fig. 5.5 (b) Generator). For the discriminator, the input will go through six CNN blocks (CNN block down 2) and then flattened to get logits (Fig. 5.5 (b) Discriminator). The descriptions and architectures of CNN block down 1, CNN block up, and CNN block down 2 are illustrated in Fig. 5.5 (b) Block descriptions.

Leaky relu activations are used in every convolutional layer except for the last layer of the generators. In contrast, the relu activation was used for the convolutional layer in the discriminators as no vanished gradient was observed for such modules.



(a)



(b)

Figure 5.5: The architecture of CycleGAN for the industrial robot data set.: (a) unconditional CycleGAN; (b) conditional CycleGAN.

(b) Hyper-parameter settings

Again, a sliced Wasserstein distance with 32 random projections ($r = 32$) was used in the loss function of the generator. The L_2 norm is also considered in the cycle consistency loss with the λ_c set to 10. The discriminator and generator learning rates were 1e-4 and 2e-5, respectively. All modules were being optimized with Adam. The maximum iterations of unconditional and conditional CycleGAN were set to 10,000 and 18,000, respectively.

5.3.3 Considered scenarios

Four scenarios were taken into consideration:

1: wd, i.e., Wasserstein distance loss in both CycleGAN and conditional CycleGAN with the neural network architecture of the ResNet;

2: wd-sem, i.e., Wasserstein distance loss in both CycleGAN and conditional CycleGAN with the neural network architecture of the modified ResNet;

3: swd, i.e., sliced Wasserstein distance loss in both CycleGAN and conditional CycleGAN with the neural network architecture of the ResNet;

4: swd-sem, i.e., sliced Wasserstein distance loss in both CycleGAN and conditional CycleGAN with the neural network architecture of the modified ResNet.

5.4 Results and discussion

5.4.1 Unconditional CycleGAN

Fig. 5.6 shows a two-dimensional visualization of the (high-dimensional) feature space using t-distributed stochastic neighbor embedding (t-SNE) [194]. In the figure, C0 (shown in red) presents the class of handwriting '0' (viewed as the source domain) while C1 (shown in green) stands for the digit '6' (viewed as the target domain). G0 stands for the generated data of class C0, while G1 denotes the generated data of class

C1. At the beginning of the training (shown in Fig. 5.6), the examples of each one of these classes are well separated. For data augmentation purposes, the goal is to generate G0 and G1 as close as possible to C0 and C1, respectively. That would correspond to a successful data transference from the source to the target domain. From this figure, it is apparent that the SWD-based CycleGAN scenarios (swd and swd-sem) are performing better when compared to the vanilla CycleGAN (scenarios wd and wd-sem). The modified residual network scenario (swd-sem) appears to outperform the others.

The same relative performance was also observed for the robot data set as illustrated in Fig. 5.7. In this figure, the number 1 (in red) stands for the nominal condition C_0. In contrast, the number 2 (in green) stands for condition C_1. Moreover, numbers 0 and 3 denote the generated data of C_0 and C_1, respectively. Again, it is apparent that SWD-based CycleGANs outperform WD-based CycleGANs.

For the sake of statistical analysis, 30 independent runs (with different random weights) were executed for each scenario. The *MC* rate is used to measure the performance of each scenario. The results obtained for each scenario are presented in the boxplots in Fig. 5.8. A boxplot consists of the min, the median (red line), and the max values, as well as outliers, of a given data sample. From this figure, one can be found that on the MNIST dataset, the median *MC* rate is i) wd: 57.86%; ii) wd-sem: 54.26%; iii) swd: 62.17%; and vi) swd-sem: 61.78%. For the industrial robot dataset, the median *MC* rate is i) wd: 93.22%; ii) wd-sem: 81.90%; iii) swd: 99.40%; and vi) swd-sem: 99.21%.

For accessing training efficiency, a threshold on the MC rate was set to check the minimum iterations required to reach such a threshold. MC thresholds of 60% for the MNIST dataset and 99.50% for the industrial robot data set were set. The results shown in Fig. 5.9 are (a). For the MNIST data set, we observed that to reach MC = 60%, the average steps for each scenario are i) wd: 427; i)wd-sem: 419; iii) swd: 125; and vi) swd-sem: 73.

For the industrial robot data set, the average number of iterations for reaching an MC of 99.5% in each scenario are i) wd: 7561; i)wd-sem: 5078; iii) swd: 130; and vi)

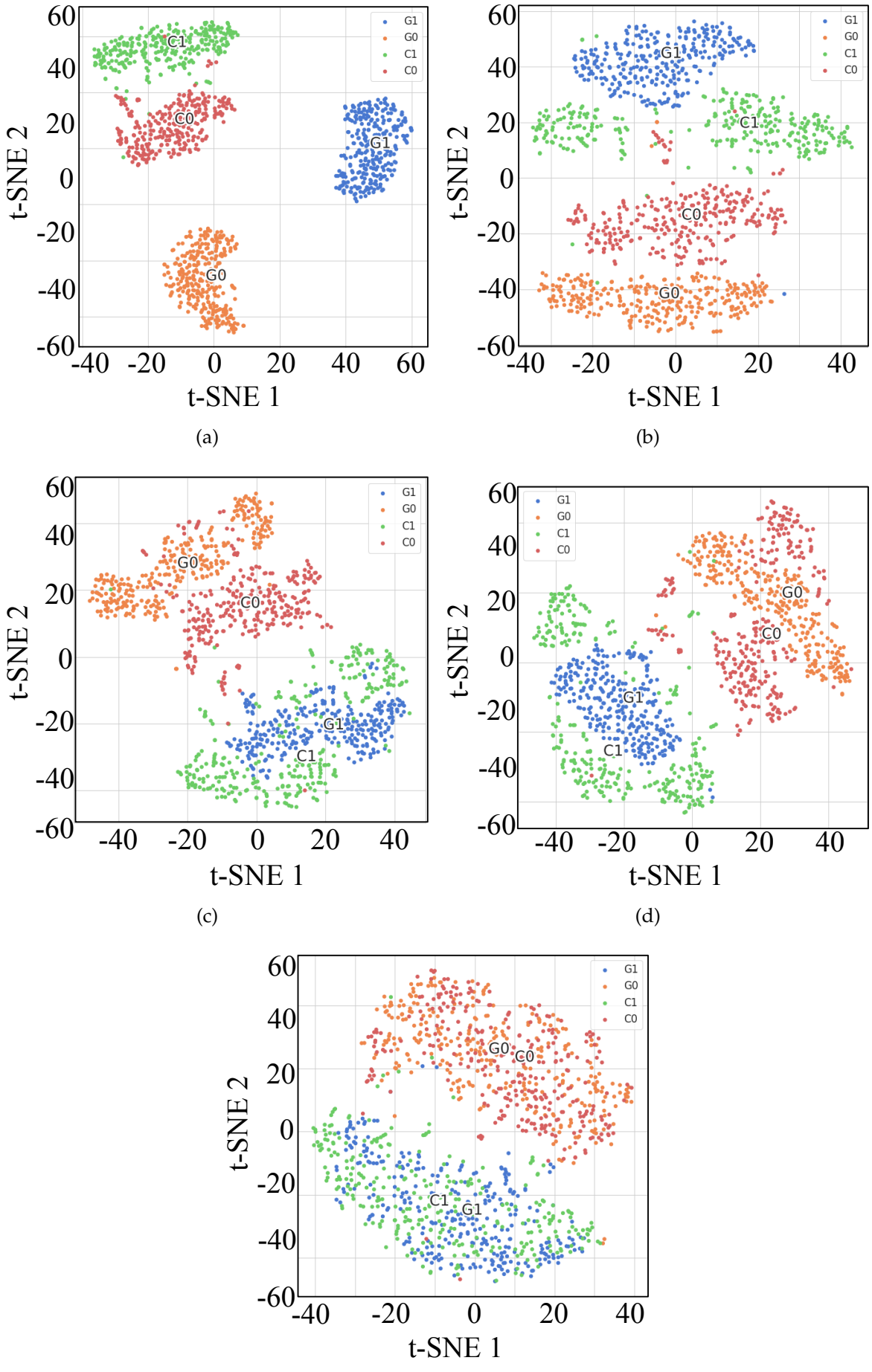
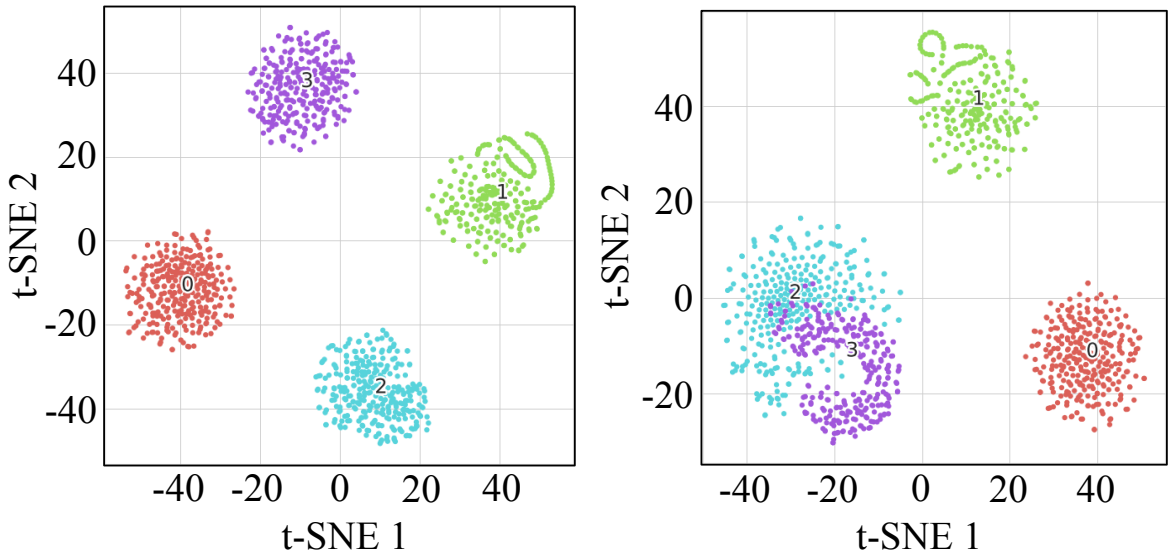
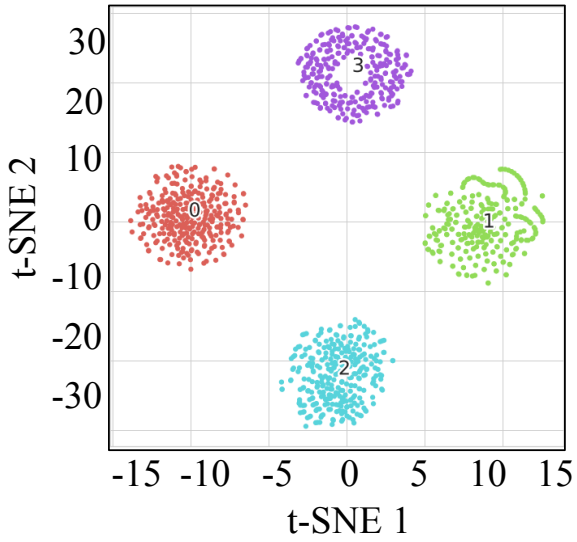


Figure 5.6: t-SNE visualization with the MNIST dataset: (a) initial state; (b) wd ;(c) wd-sem; (d) swd and (e) swd-sem.

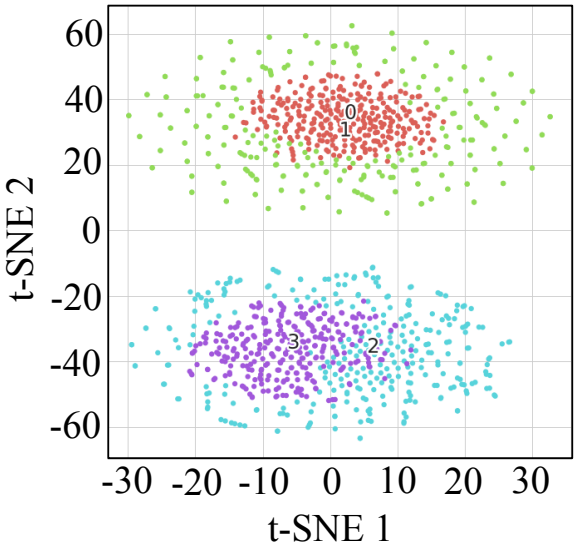


(a)

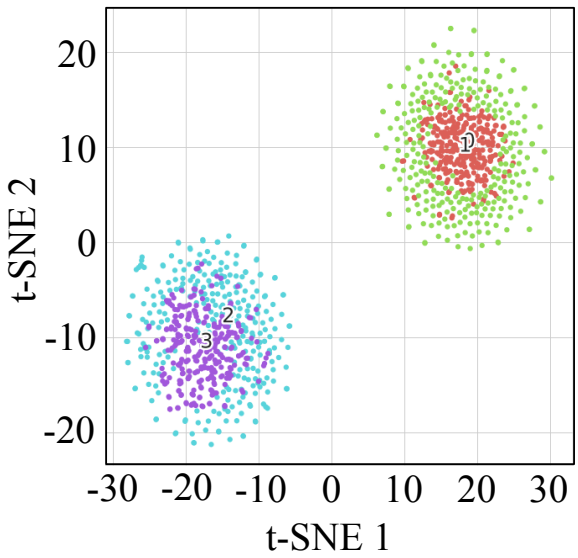
(b)



(c)

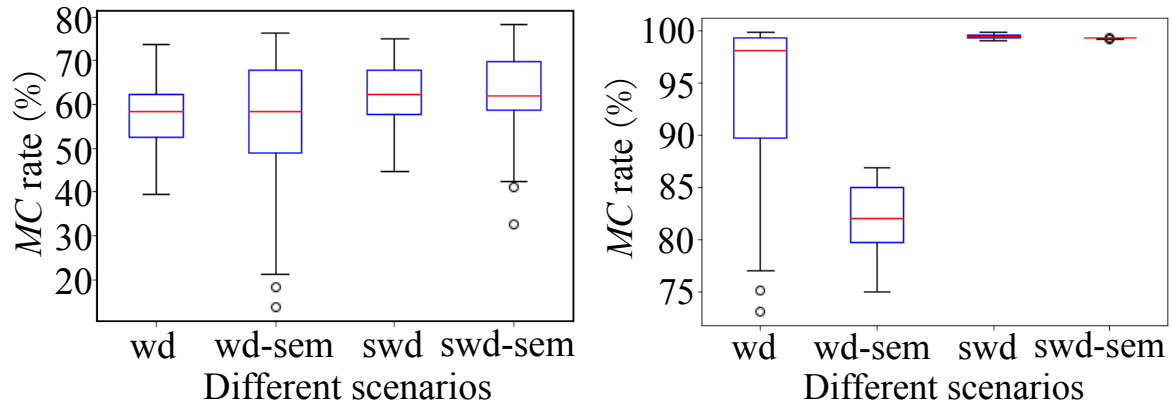


(d)



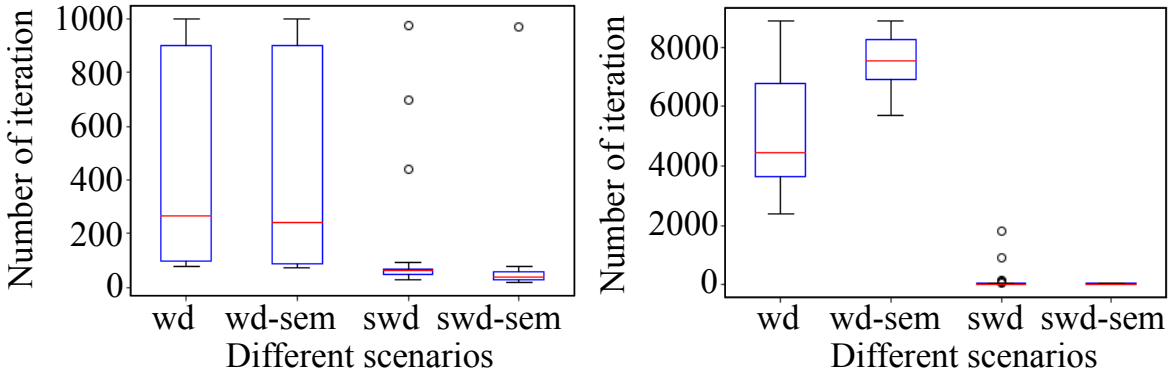
(e)

Figure 5.7: t-SNE visualization with the industrial robot dataset: (a) initial state; (b) wd ;(c) wd+sem; (d) swd and (e) swd+sem.



(a) (b)

Figure 5.8: The Boxplot of different scenarios: (a) MNIST dataset;(b) Industrial robot dataset.



(a) (b)

Figure 5.9: Dispersion of the required number of iterations to reach the same level of performance, over 30 independent runs: (a) MNIST dataset ;(b) Industrial robot.

swd-sem: 39.

This is, SWD-based models need way fewer iterations to achieve the same performance level when compared with their counterparts equipped with WD. In the case of the robot, scenario swd required 1 (one) order of magnitude less iterations than the original one (scenario wd) while scenario swd-sem required 2 (two) orders of magnitude fewer iterations than the corresponding one (wd-sem).

To quantitatively evaluate the results of Fig. 5.8, a statistical analysis was performed. The Friedman test is used for checking possible differences among the distributions. If such a difference exists, then a *post hoc* test, i.e., the Wilcoxon test, is used for ranking (see [167] for details). Friedman test yielded a $p_{\text{Friedman}} = 0.0455$ in the

Table 5.1
Wilcoxon *post-hoc* results of unconditional CycleGAN for the four studied scenarios.

Dataset	Comparison	p-value	Winner
MNIST	wd vs. wd-sem	8.16×10^{-6}	wd-sem
	wd vs. swd	5.26×10^{-6}	swd
	wd vs. swd-sem	6.94×10^{-4}	swd-sem
	wd-sem vs. swd	3.789×10^{-6}	swd
	wd-sem vs. swd-sem	3.789×10^{-6}	swd-sem
	swd vs. swd-sem	1.88×10^{-4}	swd-sem
Industrial robot	wd vs. wd-sem	8.167×10^{-5}	wd
	wd vs. swd	5.25×10^{-6}	swd
	wd vs. swd-sem	6.914×10^{-4}	swd-sem
	wd-sem vs. swd	3.78×10^{-6}	swd
	wd-sem vs. swd-sem	3.78×10^{-6}	swd-sem
	swd vs. swd-sem	1.88×10^{-4}	swd-sem

MNIST data set and a $p_{\text{Friedman}} = 2.4 \times 10^{-13}$ in the industrial robot data set, revealing that the null-hypothesis must be rejected (i.e., there is a statistically significant difference among samples) and therefore *post hocs* were ran whose results are shown in Table 5.2.

From Table 5.1 shows that scenario swd-sem has three wins for both the MNSIT and the industrial robot data sets, ranking first among other scenarios. Do this relative performance also hold for the conditional cycleGAN?

5.4.2 Conditional CycleGAN

Figs. 5.10 and 5.11 present t-SNE projection of real feature space into the plan. For the MNIST data set, the source domain is the class of digits '0' and the remaining classes of digits '1' to '9' are defined as different target domains. Notice that, in the conditional CycleGAN, the source domain is transferred to the target domain specified by an input label. In the initial state (shown in Fig. 5.10 (a)), the corresponding generated data sets (G1 to G9) are far away from the real data sets (C1 to C9). As shown in Fig 5.10 (b) to (d), denoting each one of the four above mentioned scenarios, after training, the generated data sets become closer to their corresponding real data sets, the swd-sem scenario having the largest inter-class distance among all others, as desired.

Fig. 5.11 (a) shows the generated data for G1 to G5 are distinct from the real data at

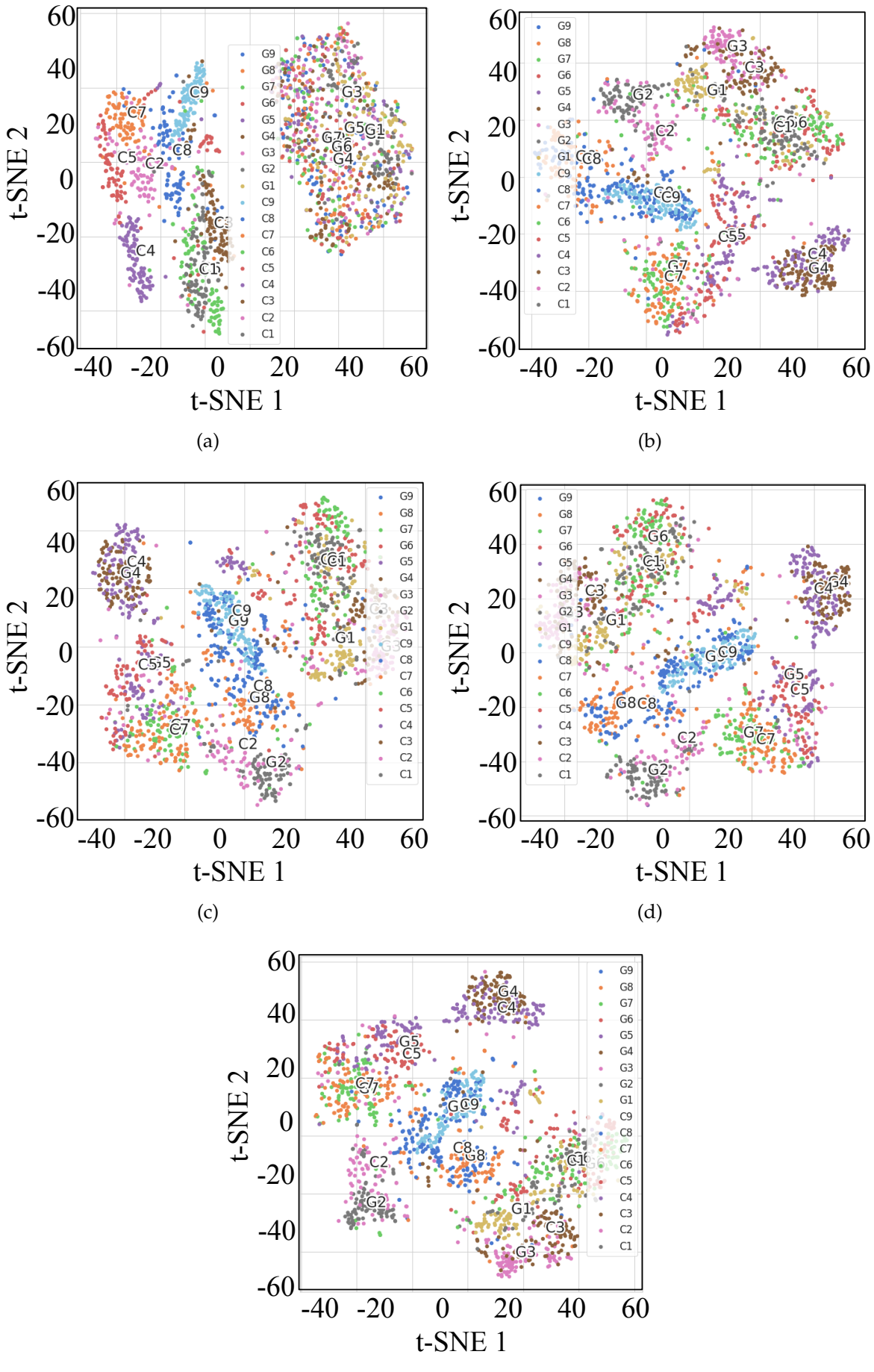


Figure 5.10: t-SNE visualization with the MNIST data set: (a) initial learning state; (b) wd ;(c) wd-sem; (d) swd and (e) swd-sem.

Table 5.2
Wilcoxon *post-hoc* results for the four studied scenarios with conditional CycleGAN .

Dataset	Comparison	p-value	Winner
MNIST	wd vs. wd-sem	4.35×10^{-4}	wd-sem
	wd vs. swd	1.36×10^{-5}	swd
	wd vs. swd-sem	3.51×10^{-6}	swd-sem
	wd-sem vs. swd	1.03×10^{-3}	swd
	wd-sem vs. swd-sem	8.46×10^{-6}	swd-sem
	swd vs. swd-sem	9.9×10^{-4}	swd-sem
Industrial robot	wd vs. wd-sem	1.4×10^{-2}	wd
	wd vs. swd	1.45×10^{-5}	swd
	wd vs. swd-sem	2.84×10^{-6}	swd-sem
	wd-sem vs. swd	2.9×10^{-4}	swd
	wd-sem vs. swd-sem	2.847×10^{-6}	swd-sem
	swd vs. swd-sem	1.7×10^{-4}	swd-sem

the beginning of the training. Fig 5.11 (b) to (e) show the obtained results after training for different scenarios. From the result presented in the figure, it is apparent that swd-sem outperforms all the others.

Fig. 5.12 shows the dispersion of the *MC* rate obtained from 30 independent runs. From this, we can find that for the MNIST data set, the median *MC* rate is i) wd: 54%; ii) wd-sem: 54.42%; iii) swd: 54.80%; and vi) swd-sem: 55.16%, and for the industrial robot i) wd: 98.96%; ii) wd-sem: 99.29%; iii) swd: 99.69%; and vi) swd-sem: 99.73%.

Since SWD has a faster convergence rate than WD in unconditional CycleGAN, considering its generalization performance, we further evaluate this phenomenon in conditional CycleGAN as shown in Fig. 5.13. A threshold of *MC* 54% for the MNIST data set and 99.5% for the robot was set. For MNIST the average number of iterations required were i) wd: 1,560; i)wd-sem: 1,466; iii swd: 1,153; and vi) swd-sem: 1,080. For the robot, the average number of iterations required were i) wd: 16,249; i)wd-sem: 15,637; iii) swd: 13,668; and vi) swd-sem: 13,034.

Again, SWD-based models needed fewer iterations than their traditional counterparts; not as notorious as the unconditional case, though.

Statistical tests of hypotheses yielded the results displayed in Table 5.2. From the above analysis, one can find that swd-sem outperforms all the others for both MNIST and the industrial robot data sets.

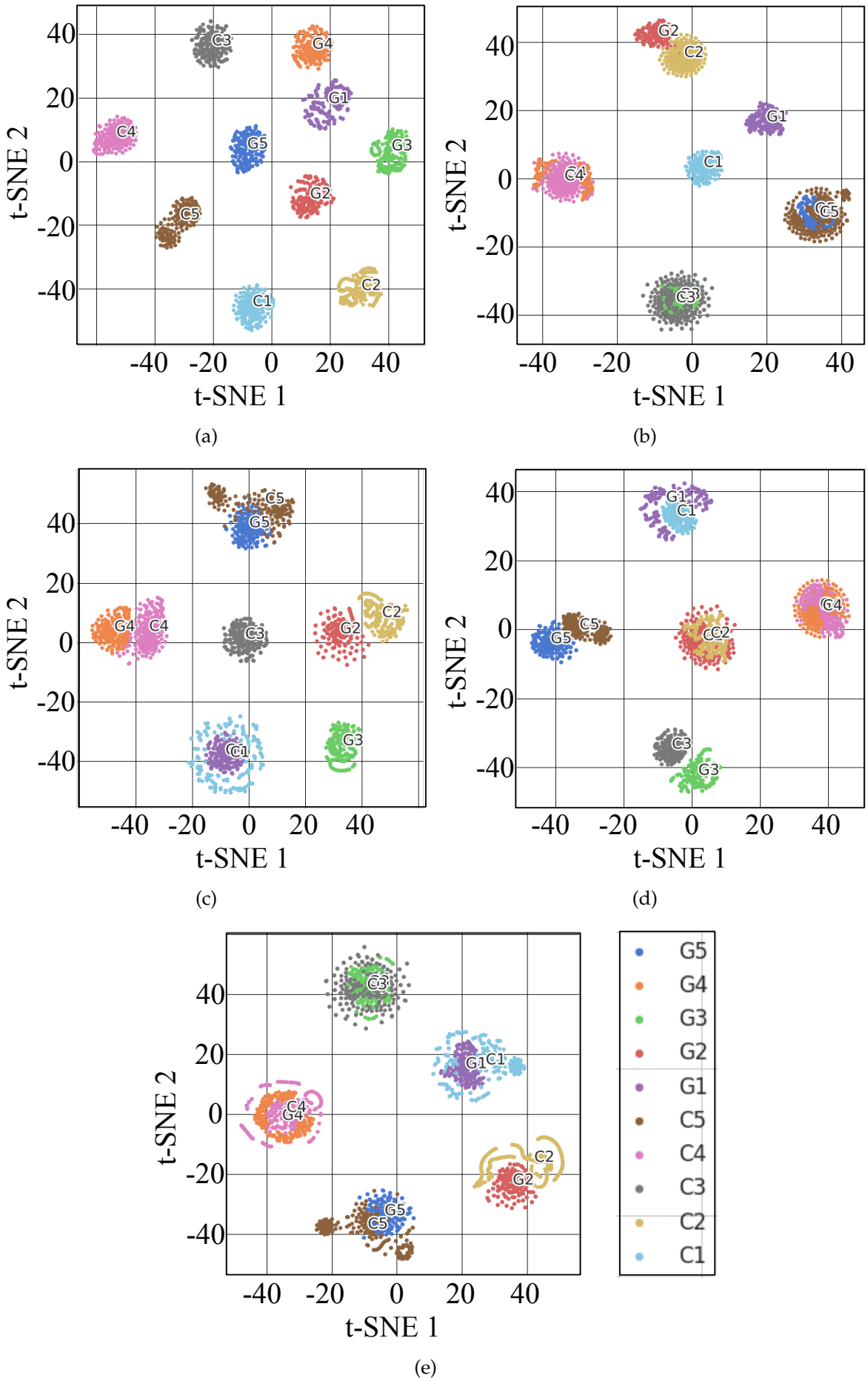
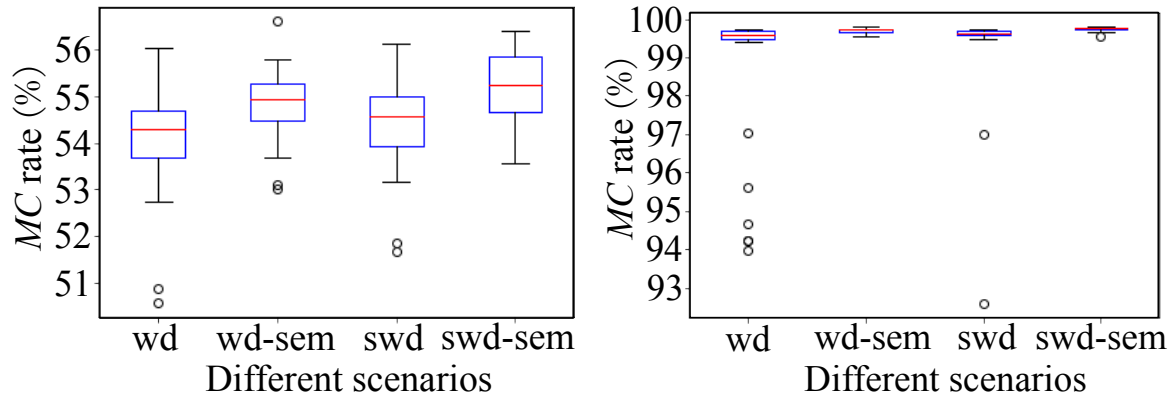
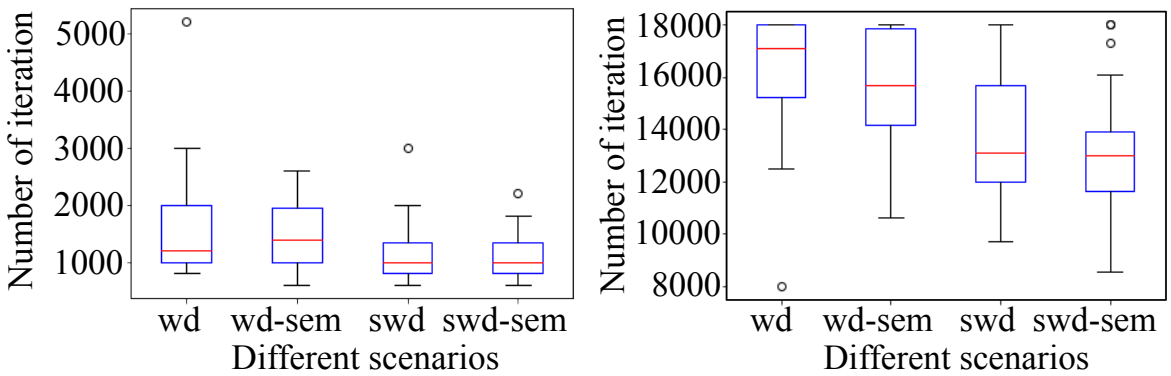


Figure 5.11: t-SNE visualization with the industrial robot data set: (a) initial learning state; (b) wd ;(c) wd-sem; (d) swd and (e) swd-sem.



(a) (b)

Figure 5.12: Boxplots for different scenarios: (a) MNIST; (b) Industrial robot.



(a) (b)

Figure 5.13: The convergence curves for the different scenarios: (a) MNIST; (b) Industrial robot.

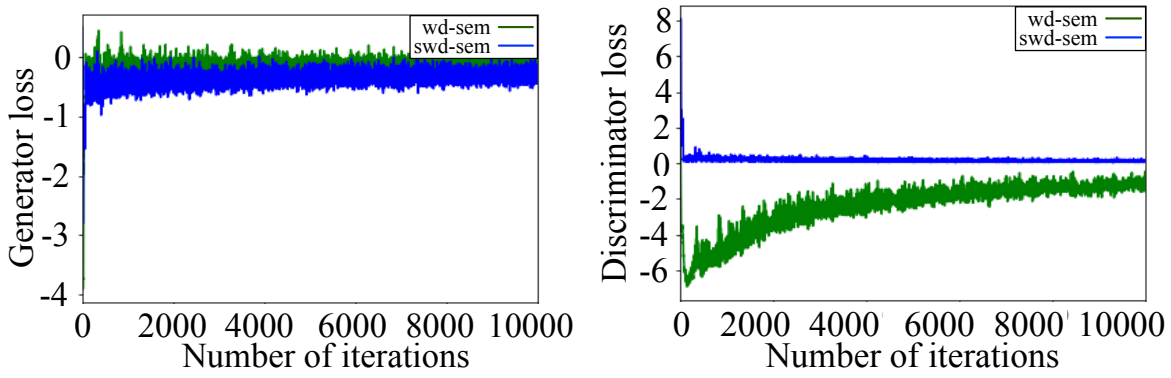


Figure 5.14: Typical convergence curve on MNIST data set for (a) Generator; (b) Discriminator.

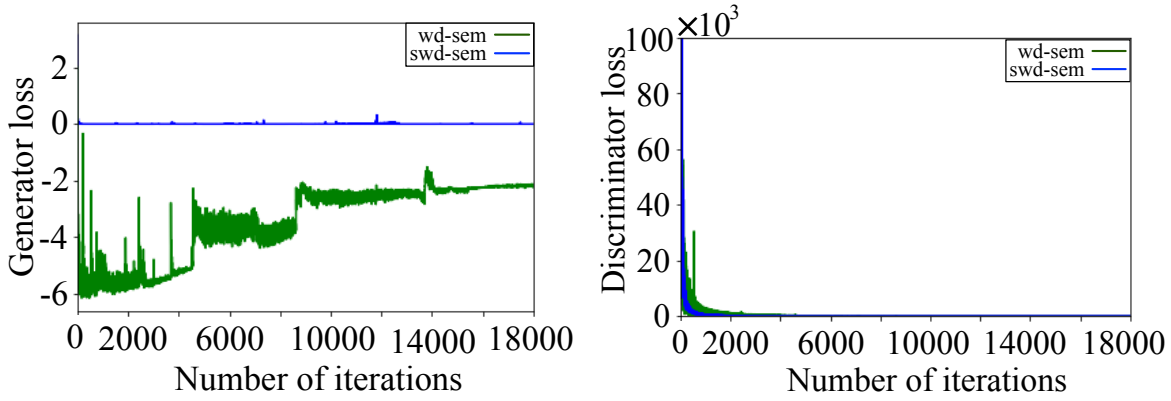


Figure 5.15: Typical convergence curve on the industrial robot data set for (a) Generator; (b) Discriminator.

Besides, convergence curves of both generator and discriminator on these two data sets are shown in Fig. 5.14 and 5.15. Here, scenarios swd-sem and wd-sem are introduced to see the phenomenon. It is clearly shown that scenario swd-sem has a smother curve than wd-sem on MNIST or the industrial robot data set.

For generating these results, we spent about 40 days in google colab with the following configurations:

- 1 GPU named Tesla V100.

- The tensorflow version is 2.3.0.

5.5 Conclusions

Generative adversarial networks (GANs) are becoming the tool of election for data generation. This is particularly so in data-driven fault diagnosis, where data from the nominal state is relatively accessible to be acquired while data from the different faulty states can be expensive to acquire, if possible. Without a balanced data set for training, fault classifiers simply do not have an acceptable out-of-sample performance.

One of the main problems in practical applications of GANs is the oscillatory behavior of loss during training and mode collapse. The loss function used plays a crucial role in the first issue. Mode collapse can be addressed by resorting to CycleGANs, a combination of two GANs used in such a way that for each input, a different synthetic example should be generated and from which the original input can be reconstructed.

With the above two premises in mind, we advocate the employment of sliced Wasserstein distance in CycleGANs, which has shown smoother, faster, and more efficient than the traditional Wasserstein CycleGAN to transfer abundant normal state data to the different scarce faulty data for an industrial robot fault diagnosis.

We based our findings on a comprehensive set of experiments over both the MNIST data set and an in-house industrial robot fault diagnosis data set. These include the unconditional and conditional versions of CycleGAN with ResNet with and without the squeeze-and-excitation mechanism. The conclusions are grounded on non-parametric statistical tests of hypotheses over the *MC* rate metric. For the unconditional case, the improvement in efficiency can be greater than 2 (two) orders of magnitude.

5.6 Appendix

This appendix specifies the used procedures. In the light of reproducible research, the python code used for generating the presented MNIST examples is publicly available from https://github.com/pzq522362451/SW-CycleGAN

6

Conclusions and Future Research

In which we present a summary of the main results of this dissertation and pinpoint some paths for future follow-ups.

6.1 Conclusions

This dissertation has presented work relevant to a new insight into the relationships between loss functions and GAN and the impact of those relationship on the GAN-training and performance. In the following, we present a summary of the main results

and deliverables from this dissertation.

6.1.1 Exploit GANs for data augmentation

The deep learning-based fault diagnosis task needs a large amount of data. However, fault data acquisition is more difficult in the mechanical system. Consequently, an imbalanced data set can be obtained that cannot ensure good performance on the intelligent fault diagnosis. Therefore, it is necessary to acquire enough faulty data. With some data augmentation techniques like SMOTE or VAE, researchers have the ability to obtain a large dataset. In this work, we utilize GANs for data augmentation. The results from this part of the work derive from our attempt to answer the following questions:

- *Can GAN replace oversampling techniques such as SMOTE for imbalanced data augmentation in the industrial field?*

In this context, we investigate GAN as an oversampling technique for data augmentation. Generally, a WPT combined with GAN is used for feature generation while an RF is used for fault classification. The features were firstly calculated using WPT, then the acquired features were sent to WGAN for data generation. Finally, with a balanced data set, a classification task is performed to show the feasibility of GAN for data augmentation.

We performed a comprehensive evaluation of the impact of this framework on several perspectives with different scenarios for comparison. The perspective, such as generator selection, the number of training examples in each class, training data shuffling, the distribution used for sampling input random data, and initial conditions, are well discussed. Interestingly enough, the obtained experimental results show that the proposed framework can be viewed as an oversampling tool in terms of data augmentation.

An additional argument supporting the effectiveness of our approach comes from that it was possible to increase the performance of the fault diagnosis for an industrial

robot for any of the GANs-based models over classical undersampling and oversampling (SMOTE) methods. Besides, in many cases like prognostics and health management, without enough data set, the performance of the monitoring capability of the industrial system will be decreased. GAN is an efficient tool to get rid of the limitation of data imbalance state, which can enhance the monitoring capability in PHM.

6.1.2 Developing GANs with V-matrix based loss function

Due to the adversarial learning between the discriminator and the generator in GAN, the loss curve is not steady which makes GAN training become oscillating. It is still difficult to train them to obtain a generator with the desired performance. Different from traditional deep learning-based models where the lower loss tends to appear in the last iteration, GAN cannot ensure the best value at the end of training. To this end, we performed a study to address the training oscillation problem from a perspective of the loss function and pose the following research question:

- *Can V-matrix be used as the loss function of GAN to address training oscillation?*

Besides, traditional GAN can learn the data distribution from a limited faulty data class where there are many kinds of faulty data classes in the industrial system. For the new faulty data class, it needs to be sent to this model to learn the new distribution again to generate enough data that will bring time cost for training. Thus, we also pose the question:

- *Can GAN generates different kinds of faulty data classes at one time in the industrial field?*

To answer these two questions, in this work, we propose a generalization of both mean square error (MSE) GAN and Wasserstein GAN (WGAN) with gradient penalty, referred to as VGAN. Within conditional WGAN with gradient penalty, VGAN resorts to the Vapnik V-matrix-based criterion that generalizes MSE. Also, a novel stop criterion is proposed to keep track of the most suitable model during training.

The proposed algorithm is tested with different scenarios through a comprehensive set of experiments on both the MNIST case and a fault-diagnosis task for an industrial robot where the generative model is used as a data augmentation tool for dealing with imbalanced datasets is presented. The statistical analysis of the results demonstrates that the proposed model in this study outperforms other models, including vanilla GAN, conditional WGAN with and without conventional regularization, and synthetic minority oversampling technique, a classic data augmentation technique.

6.1.3 Considering the Sliced Wasserstein distance on CycleGAN

In this context, we aimed at answering the following:

- *Can the sliced Wasserstein distance be used in GAN for data estimation?*
- *Can the sliced Wasserstein distance be better than the Wasserstein distance for data transfer using CycleGAN?*
- *Can we use a conditional version of CycleGAN for data transfer to enhance the training efficiency?*

In this regard, we propose CycleGAN with the sliced Wasserstein distance (SWD) to transfer abundant normal data of the industrial robot to different scarce fault data. In this work, we evaluate the feasibility of SWD from unconditional CycleGAN to the conditional CycleGAN with and without squeeze-and-excitation mechanisms.

We believe that the point of departure is worthy since we found further developments in the training stabilization of GAN. We traduced SWD, which is usually for the measurement of two distributions distance into the loss function of CycleGAN, and demonstrated its generalization ability and feasibility via a comprehensive set of experiments.

The results are also encouraging because, in the proposed experiments, the proposed algorithm converges faster and is more stable than using the Wasserstein distance. More importantly, this algorithm in the conditional state can transfer normal

data to different fault data by sending desired label information. In addition, there is no need to re-train the model several times, which reduces the computational cost.

In the following section we draw some lines for further developments regarding Chapters 3, 4 and 5.

6.2 Future Research

GAN currently holds the state-of-the-art on most image generation tasks according to some evaluation metrics like FID, IS, or MS-SSIM. However, those metrics do not fully capture diversities, and there is a report showing that GAN captures less diversity of data distribution than likelihood-based models. With three main problems on GAN, it is very challenging to be well trained. This dissertation leaves some open questions that we hope can provide a foothold for further developments.

1. The results in Chapter 3 state that the application of GANs for the data augmentation of the industrial robot is very promising and indicates that the proposed framework is plausible. Hence we can use GAN as the oversampling tool to satisfy the data need for the intelligent fault diagnosis. However, the time series is transformed with Wavelet packet transform with features calculation. We would like to raise the opportunity to answer "*Can GAN has the ability to generate the raw time series signals?*"

2. Considering the training instability problem in GAN, Chapter 4 used a V-matrix based loss function as a regularization on WGAN to improve the generation performance. The GANs we used here is the conditional version of GANs that can generate desired outputs. In this setting, the following question seems pertinent "*Can we show the theoretical convergence under V-matrix based losses?*"

3. Wasserstein GAN has shown remarkable progress in GANs' research. In Chapter 5, we consider an alternative loss function named the Sliced Wasserstein distance for GAN for transferring data from abundant normal state data to different scarce faulty data. So we pose the following question "*Can the sliced Wasserstein distance completely capture the data distribution in any GANs model?*"

Drawbacks in GAN make it challenging to scale and apply to new domains. In [195, 196, 197], they use likelihood-based models to achieve GAN-like sample quality except for VAE. Thus the likelihood-based models capture more diversity and are easy to train than GAN.

With this in mind, the diffusion model [198] is a class of likelihood-based models that have been shown to generate high-quality samples. It shows easy scalability, distribution convergence, and, importantly, a stationary training process than GAN.

Bibliography

[1] Pawel Andrzej Laski and Mateusz Smykowski. Using a development platform with an stm32 processor to prototype an inexpensive 4-dof delta parallel robot. *Sensors*, 21(23), 2021.

[2] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017.

[3] Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10285–10295, 2019.

[4] Jianyu Long, Yaoxin Qin, Zhe Yang, Yunwei Huang, and Chuan Li. Discriminative feature learning using a multiscale convolutional capsule network from attitude data for fault diagnosis of industrial robots. *Mechanical Systems and Signal Processing*, 182:109569, 2023.

[5] Ola Pettersson. Execution monitoring in robotics: A survey. *Robotics and Autonomous Systems*, 53(2):73–88, 2005.

[6] Bouchra Abouelanouar, Mostafa Elamrani, Bachir Elkihel, and Fabienne Delaunois. Application of wavelet analysis and its interpretation in rotating machines monitoring and fault diagnosis. a review. *Int J Eng Technol*, 7(4):3465–3471, 2018.

[7] Yiyuan Gao, Dejie Yu, and Haojiang Wang. Fault diagnosis of rolling bearings using weighted horizontal visibility graph and graph fourier transform. *Measurement*, 149:107036, 2020.

[8] Georgios Mademlis, Nimananda Sharma, Yujing Liu, and Junfei Tang. Zero-sequence current reduction technique for electrical machine emulators with dc-coupling by regulating the svm zero states. *IEEE Transactions on Industrial Electronics*, 2021.

[9] Hui Zhang, Ping Chen, and Qiang Wang. Fault diagnosis method based on eemd and multi-class logistic regression. In *2018 3rd International Conference on Smart City and Systems Engineering (ICSCSE)*, pages 859–863. IEEE, 2018.

[10] Wanke Yu and Chunhui Zhao. Online fault diagnosis for industrial processes with bayesian network-based probabilistic ensemble learning strategy. *IEEE Transactions on Automation Science and Engineering*, 16(4):1922–1932, 2019.

[11] Imad Abdallah, V Dertimanis, H Mylonas, Konstantinos Tatsis, Eleni Chatzi, N Dervili, K Worden, and Eoghan Maguire. Fault diagnosis of wind turbine structures using decision tree learning algorithms with big data. In *Safety and Reliability–Safe Societies in a Changing World*, pages 3053–3061. CRC Press, 2018.

[12] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Abubakar Malah Umar, Okafor Uchenwa Linus, Humaira Arshad, Abdullahi Aminu Kazaure, Usman Gana, and Muhammad Ubale Kiru. Comprehensive review of artificial neural network applications to pattern recognition. *IEEE Access*, 7:158820–158846, 2019.

[13] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[14] Feng Jia, Yaguo Lei, Liang Guo, Jing Lin, and Saibo Xing. A neural network constructed by deep learning technique and its application to intelligent fault diagnosis of machines. *Neurocomputing*, 272:619–628, 2018.

[15] Zhuyun Chen and Weihua Li. Multisensor feature fusion for bearing fault diagnosis using sparse autoencoder and deep belief network. *IEEE Transactions on Instrumentation and Measurement*, 66(7):1693–1702, 2017.

[16] Wenfeng Gong, Yuanzhe Wang, Meiling Zhang, Ehsan Mihankhah, Hui Chen, and Danwei Wang. A fast anomaly diagnosis approach based on modified cnn and multi-sensor data fusion. *IEEE Transactions on Industrial Electronics*, 2021.

[17] Jianyu Long, Shaohui Zhang, and Chuan Li. Evolving deep echo state networks for intelligent fault diagnosis. *IEEE Transactions on Industrial Informatics*, 16(7):4928–4937, 2019.

[18] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[19] Abdel-rahman Mohamed, George E Dahl, and Geoffrey Hinton. Acoustic modeling using deep belief networks. *IEEE transactions on audio, speech, and language processing*, 20(1):14–22, 2011.

[20] Nitish Srivastava and Russ R Salakhutdinov. Multimodal learning with deep boltzmann machines. *Advances in neural information processing systems*, 25, 2012.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[22] Willem E ávan der Linden et al. Tutorial review—data processing by neural networks in quantitative chemical analysis. *Analyst*, 118(4):323–328, 1993.

[23] Aliaksei Sandryhaila and José MF Moura. Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure. *IEEE Signal Processing Magazine*, 31(5):80–90, 2014.

[24] S Joe Qin. Process data analytics in the era of big data. *AIChE Journal*, 60(9):3092–3100, 2014.

[25] Konstantinos Makantasis, Konstantinos Karantzalos, Anastasios Doulamis, and Nikolaos Doulamis. Deep supervised learning for hyperspectral data classification through convolutional neural networks. In *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 4959–4962. IEEE, 2015.

[26] Yuyan Zhang, Xinyu Li, Liang Gao, Lihui Wang, and Long Wen. Imbalanced data fault diagnosis of rotating machinery using synthetic oversampling and feature learning. *Journal of manufacturing systems*, 48:34–50, 2018.

[27] MingHong Han, Yaman Wu, Yunfeng Huang, and Yumin Wang. A fault diagnosis method based on improved synthetic minority oversampling technique and svm for unbalanced data. In *IOP Conference Series: Materials Science and Engineering*, volume 1043, page 052034. IOP Publishing, 2021.

[28] Kun Yu, Tian Ran Lin, Hui Ma, Xiang Li, and Xu Li. A multi-stage semi-supervised learning approach for intelligent fault diagnosis of rolling bearing using data augmentation and metric learning. *Mechanical Systems and Signal Processing*, 146:107043, 2021.

[29] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[30] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *Computing Research Repository*, page arXiv:1411.1784, 2014.

[31] Christian Ledig, Lucas Theis, Ferenc Huszár, José Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[32] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[33] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[34] Funa Zhou, Shuai Yang, Hamido Fujita, Danmin Chen, and Chenglin Wen. Deep learning fault diagnosis method based on global optimization gan for unbalanced data. *Knowledge-Based Systems*, 187:104837, 2020.

[35] Ziqiang Pu, Diego Cabrera, René-Vinicio Sánchez, Mariela Cerrada, Chuan Li, and José Valente de Oliveira. Exploiting generative adversarial networks as an oversampling method for fault diagnosis of an industrial robotic manipulator. *Applied Sciences*, 10(21):7712, 2020.

[36] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

[37] Ziqiang Pu, Diego Cabrera, René-Vinicio Sánchez, Mariela Cerrada, Chuan Li, and José Valente de Oliveira. Exploiting generative adversarial networks as an oversampling method for fault diagnosis of an industrial robotic manipulator. *Applied Sciences*, 10(21), 2020.

[38] Vladimir Vapnik and Rauf Izmailov. V-matrix method of solving statistical inference problems. *J. Mach. Learn. Res.*, 16(51):1683–1730, 2015.

[39] Vladimir Vapnik and Rauf Izmailov. Rethinking statistical learning theory: learning using statistical invariants. *Machine Learning*, 108(3):381–423, 2019.

[40] Shan Yang, Lei Xie, Xiao Chen, Xiaoyan Lou, Xuan Zhu, Dongyan Huang, and Haizhou Li. Statistical parametric speech synthesis using generative adversarial networks under a multi-task learning framework. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 685–691. IEEE, 2017.

[41] Ziqiang Pu, Diego Cabrera, Chuan Li, and José Valente de Oliveira. Vgan: Generalizing mse gan and wgan-gp for robot fault diagnosis. *IEEE Intelligent Systems*, pages 1–1, 2022.

[42] Sliced wasserstein cycle consistency generative adversarial networks for fault data augmentation of an industrial robot. *Expert Systems with Applications*, 222:119754, 2023.

[43] Junjun Zhu, Quansheng Jiang, Yehu Shen, Chenhui Qian, Fengyu Xu, and Qixin Zhu. Application of recurrent neural network to mechanical fault diagnosis: A review. *Journal of Mechanical Science and Technology*, pages 1–16, 2022.

[44] S Manikandan and K Duraivelu. Fault diagnosis of various rotating equipment using machine learning approaches–a review. *Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering*, 235(2):629–642, 2021.

[45] Haoqiang Liu, Hongbo Zhao, Jiayue Wang, Shuai Yuan, and Wenquan Feng. Lstm-gan-ae: A promising approach for fault diagnosis in machine health monitoring. *IEEE Transactions on Instrumentation and Measurement*, 71:1–13, 2021.

[46] Qiao Xue, Guang Li, Yuanjian Zhang, Shiquan Shen, Zheng Chen, and Yonggang Liu. Fault diagnosis and abnormality detection of lithium-ion battery packs based on statistical distribution. *Journal of Power Sources*, 482:228964, 2021.

[47] Jinde Zheng, Miaoxian Su, Wanming Ying, Jinyu Tong, and Ziwei Pan. Improved uniform phase empirical mode decomposition and its application in machinery fault diagnosis. *Measurement*, 179:109425, 2021.

[48] Adam Glowacz. Fault diagnosis of electric impact drills using thermal imaging. *Measurement*, 171:108815, 2021.

[49] Kun Zhang, Chaoyong Ma, Yonggang Xu, Peng Chen, and Jianxi Du. Feature extraction method based on adaptive and concise empirical wavelet transform and its applications in bearing fault diagnosis. *Measurement*, 172:108976, 2021.

[50] Hao Wu, Xue Ma, and Chenglin Wen. Multilevel fine fault diagnosis method for motors based on feature extraction of fractional fourier transform. *Sensors*, 22(4):1310, 2022.

[51] Yuguo Zhou, Shaoting Yan, Yanbo Ren, and Shiliang Liu. Rolling bearing fault diagnosis using transient-extracting transform and linear discriminant analysis. *Measurement*, 178:109298, 2021.

[52] Van-Cuong Nguyen, Duy-Tang Hoang, Xuan-Toa Tran, Mien Van, and Hee-Jun Kang. A bearing fault diagnosis method using multi-branch deep neural network. *Machines*, 9(12):345, 2021.

[53] Te Han, Chao Liu, Rui Wu, and Dongxiang Jiang. Deep transfer learning with limited data for machinery fault diagnosis. *Applied Soft Computing*, 103:107150, 2021.

[54] Tongyang Pan, Jinglong Chen, Jun Pan, and Zitong Zhou. A deep learning network via shunt-wound restricted boltzmann machines using raw data for fault detection. *IEEE Transactions on Instrumentation and Measurement*, 69(7):4852–4862, 2019.

[55] Shengnan Tang, Shouqi Yuan, and Yong Zhu. Data preprocessing techniques in convolutional neural network based on fault diagnosis towards rotating machinery. *IEEE Access*, 8:149487–149496, 2020.

[56] Gurkan Aydemir and Burak Acar. Anomaly monitoring improves remaining useful life estimation of industrial machinery. *Journal of Manufacturing Systems*, 56:463–469, 2020.

[57] Kwangsuk Lee, Jae-Kyeong Kim, Jaehyong Kim, Kyeon Hur, and Hagbae Kim. Cnn and gru combination scheme for bearing anomaly detection in rotating machinery health monitoring. In *2018 1st IEEE International conference on knowledge innovation and invention (ICKII)*, pages 102–105. IEEE, 2018.

[58] Harsh S Dhiman, Dipankar Deb, SM Muyeen, and Innocent Kamwa. Wind turbine gearbox anomaly detection based on adaptive threshold and twin support vector machines. *IEEE Transactions on Energy Conversion*, 36(4):3462–3469, 2021.

[59] Jianpeng Ma, Chengwei Li, and Guangzhu Zhang. Rolling bearing fault diagnosis based on deep learning and autoencoder information fusion. *Symmetry*, 14(1):13, 2021.

[60] Yang Wang, Miaomiao Yang, Yong Li, Zeda Xu, Jie Wang, and Xia Fang. A multi-input and multi-task convolutional neural network for fault diagnosis based on bearing vibration signal. *IEEE Sensors Journal*, 21(9):10946–10956, 2021.

[61] Wenliao Du, Zhen Guo, Chuan Li, Xiaoyun Gong, and Ziqiang Pu. From anomaly detection to novel fault discrimination for wind turbine gearboxes with

a sparse isolation encoding forest. *IEEE Transactions on Instrumentation and Measurement*, 71:1–10, 2022.

[62] Ying Zheng, Wei Zhou, Weidong Yang, Lang Liu, Yuanle Liu, and Yong Zhang. Multivariate/minor fault diagnosis with severity level based on bayesian decision theory and multidimensional rbc. *Journal of Process Control*, 101:68–77, 2021.

[63] Ziqiang Pu, Chuan Li, Shaohui Zhang, and Yun Bai. Fault diagnosis for wind turbine gearboxes by using deep enhanced fusion network. *IEEE Transactions on Instrumentation and Measurement*, 70:1–11, 2020.

[64] Anh-Duc Pham and Hyeong-Joon Ahn. High precision reducers for industrial robots driving 4th industrial revolution: state of arts, analysis, design, performance evaluation and perspective. *International journal of precision engineering and manufacturing-green technology*, 5(4):519–533, 2018.

[65] Alessandro Gasparetto and Lorenzo Scalera. A brief history of industrial robotics in the 20th century. *Advances in Historical Studies*, 8(1):24–35, 2019.

[66] Meng Fanzhao, Zhao Hongxia, and Wei Dongpo. Design of a three-degree-of-freedom redundant drive parallel robot. In *Journal of Physics: Conference Series*, volume 1676, page 012201. IOP Publishing, 2020.

[67] Xiaolin Zhang, Peng Han, Li Xu, Fei Zhang, Yongping Wang, and Lu Gao. Research on bearing fault diagnosis of wind turbine gearbox based on 1dcnn-pso-svm. *IEEE Access*, 8:192248–192258, 2020.

[68] Zhaokun Zhang, Guangqiang Xie, Zhufeng Shao, and Clément Gosselin. Kinematic calibration of cable-driven parallel robots considering the pulley kinematics. *Mechanism and Machine Theory*, 169:104648, 2022.

[69] Hexu Yang, Xiaopeng Li, Jinchi Xu, Yajing Guo, and Baitao Li. Modeling and fatigue characteristic analysis of the gear flexspline of a harmonic reducer. *Mathematics*, 10(6):868, 2022.

[70] Yuanchang Lin, Wencheng Sun, Guotian He, and Zhenjun Zhang. Overview of robotic reducer testing technology. *Journal of Physics: Conference Series*, 2002(1):012025, aug 2021.

[71] Izaz Raouf, Hyewon Lee, and Heung Soo Kim. Mechanical fault detection based on machine learning for robotic rv reducer using electrical current signature analysis: a data-driven approach. *Journal of Computational Design and Engineering*, 9(2):417–433, 2022.

[72] Ting Ye and Lihong Zhao. Design of industrial robot teaching system based on machine vision. In *2021 IEEE 5th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC)*, volume 5, pages 279–284, 2021.

[73] Weihua Li, Ruyi Huang, Jipu Li, Yixiao Liao, Zhuyun Chen, Guolin He, Ruqiang Yan, and Konstantinos Gryllias. A perspective survey on deep transfer learning for fault diagnosis in industrial scenarios: Theories, applications and challenges. *Mechanical Systems and Signal Processing*, 167:108487, 2022.

[74] Chuanxia Jian, Kaijun Yang, and Yinhui Ao. Industrial fault diagnosis based on active learning and semi-supervised learning using small training set. *Engineering Applications of Artificial Intelligence*, 104:104365, 2021.

[75] Tianci Zhang, Jinglong Chen, Fudong Li, Kaiyu Zhang, Haixin Lv, Shuilong He, and Enyong Xu. Intelligent fault diagnosis of machines with small and imbalanced data: A state-of-the-art review and possible extensions. *ISA Transactions*, 119:152–171, 2022.

[76] Shaowei Liu, Hongkai Jiang, Zhenghong Wu, and Xingqiu Li. Data synthesis using deep feature enhanced generative adversarial networks for rolling bearing imbalanced fault diagnosis. *Mechanical Systems and Signal Processing*, 163:108139, 2022.

[77] Jungang Xu, Hui Li, and Shilong Zhou. An overview of deep generative models. *IETE Technical Review*, 32(2):131–139, 2015.

[78] Aurélien Decelle and Cyril Furtlehner. Restricted boltzmann machine: Recent advances and mean-field theory. *Chinese Physics B*, 30(4):040202, 2021.

[79] Yihui Xiong and Renguang Zuo. Robust feature extraction for geochemical anomaly recognition using a stacked convolutional denoising autoencoder. *Mathematical Geosciences*, 54(3):623–644, 2022.

[80] Jaehyeon Kim, Jungil Kong, and Juhee Son. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*, pages 5530–5540. PMLR, 2021.

[81] Hojjat Navidan, Parisa Fard Moshiri, Mohammad Nabati, Reza Shahbazian, Seyed Ali Ghorashi, Vahid Shah-Mansouri, and David Windridge. Generative adversarial networks (gans) in networking: A comprehensive survey & evaluation. *Computer Networks*, 194:108149, 2021.

[82] Sixin Zhang. On the nash equilibrium of moment-matching gans for stationary gaussian processes. *ArXiv*, abs/2203.07136, 2022.

[83] M Premkumar, Pradeep Jangir, C Ramakrishnan, G Nalinipriya, Hassan Haes Alhelou, and B Santhosh Kumar. Identification of solar photovoltaic model parameters using an improved gradient-based optimization algorithm with chaotic drifts. *IEEE Access*, 9:62347–62379, 2021.

[84] Slawomir Koziel and Anna Pietrenko-Dabrowska. Accelerated gradient-based optimization of antenna structures using multifidelity simulations and convergence-based model management scheme. *IEEE Transactions on Antennas and Propagation*, 69(12):8778–8789, 2021.

[85] Chuan Li, Diego Cabrera, Fernando Sancho, René-Vinicio Sánchez, Mariela Cerrada, Jianyu Long, and José Valente de Oliveira. Fusing convolutional generative adversarial encoders for 3d printer fault detection with only normal condition signals. *Mechanical Systems and Signal Processing*, 147:107108.

[86] Wentao Mao, Yamin Liu, Ling Ding, and Yuan Li. Imbalanced fault diagnosis of rolling bearing based on generative adversarial network: A comparative study. *IEEE Access*, 7:9515–9530, 2019.

[87] Ziqiang Pu, Diego Cabrera, Yun Bai, and Chuan Li. A one-class generative adversarial detection framework for multifunctional fault diagnoses. *IEEE Transactions on Industrial Electronics*, 69(8):8411–8419, 2022.

[88] Wenqian Jiang, Yang Hong, Beitong Zhou, Xin He, and Cheng Cheng. A gan-based anomaly detection approach for imbalanced industrial time series. *IEEE Access*, 7:143608–143619, 2019.

[89] Chuan Li, Diego Cabrera, Fernando Sancho, René-Vinicio Sánchez, Mariela Cerrada, and José Valente de Oliveira. One-shot fault diagnosis of 3d printers through improved feature space learning. *IEEE Trans. On Industrial Electronics*, 147:107108.

[90] You-ren Wang, Guo-dong Sun, and Qi Jin. Imbalanced sample fault diagnosis of rotating machinery using conditional variational auto-encoder generative adversarial network. *Applied Soft Computing*, page 106333, 2020.

[91] Zhirui Zhang, Shujie Liu, Mu Li, Ming Zhou, and Enhong Chen. Bidirectional generative adversarial networks for neural machine translation. In *Proceedings of the 22nd conference on computational natural language learning*, pages 190–199, 2018.

[92] Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. Improving neural machine translation with conditional sequence generative adversarial nets. *arXiv preprint arXiv:1703.04887*, 2017.

[93] Baojie Li, Claude Delpha, Demba Diallo, and A Migan-Dubois. Application of artificial neural networks to photovoltaic fault detection and diagnosis: A review. *Renewable and Sustainable Energy Reviews*, 138:110512, 2021.

[94] Jasir Jawad, Alaa H Hawari, and Syed Javaid Zaidi. Artificial neural network modeling of wastewater treatment and desalination using membrane processes: A review. *Chemical Engineering Journal*, 419:129540, 2021.

[95] Victor Costa, Nuno Lourenço, João Correia, and Penousal Machado. Exploring the evolution of gans through quality diversity. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, pages 297–305, 2020.

[96] Ian J. Goodfellow. NIPS 2016 tutorial: Generative adversarial networks. *CoRR*, abs/1701.00160, 2017.

[97] Jonas Adler and Sebastian Lunz. Banach wasserstein gan. *Advances in neural information processing systems*, 31, 2018.

[98] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[99] David Warde-Farley and Yoshua Bengio. Improving generative adversarial networks with denoising feature matching. In *ICLR*, 2017.

[100] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[101] J.T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR (workshop track)*, 2015.

[102] Xudong Mao, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2813–2821, 2017.

[103] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[104] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[105] Jinrui Wang, Baokun Han, Huaiqian Bao, Mingyan Wang, Zhenyun Chu, and Yuwei Shen. Data augment method for machine fault diagnosis using conditional generative adversarial networks. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 234(12):2719–2727, 2020.

[106] Yogesh Balaji, Martin Renqiang Min, Bing Bai, Rama Chellappa, and Hans Peter Graf. Conditional gan with discriminative filter generation for text-to-video synthesis. In *IJCAI*, volume 1, page 2, 2019.

[107] Tao Hu, Chengjiang Long, and Chunxia Xiao. A novel visual representation on text using diverse conditional gan for visual recognition. *IEEE Transactions on Image Processing*, 30:3499–3512, 2021.

[108] Ayushman Dash, John Cristian Borges Gamboa, Sheraz Ahmed, Marcus Liwicki, and Muhammad Zeshan Afzal. Tac-gan - text conditioned auxiliary classifier generative adversarial network. *CoRR*, abs/1703.06412, 2017.

[109] Jon Gauthier. Conditional generative adversarial nets for convolutional face generation. *Class project for Stanford CS231N: convolutional neural networks for visual recognition, Winter semester*, 2014(5):2, 2014.

[110] Shuyang Gu, Jianmin Bao, Hao Yang, Dong Chen, Fang Wen, and Lu Yuan. Mask-guided portrait editing with conditional gans. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3436–3445, 2019.

[111] Hao Tang, Dan Xu, Nicu Sebe, Yanzhi Wang, Jason J Corso, and Yan Yan. Multi-channel attention selection gan with cascaded semantic guidance for cross-view image translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2417–2426, 2019.

[112] Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. Towards diverse and natural image descriptions via a conditional gan. In *Proceedings of the IEEE international conference on computer vision*, pages 2970–2979, 2017.

[113] Shunyu Yao, Tzu Ming Hsu, Jun-Yan Zhu, Jiajun Wu, Antonio Torralba, Bill Freeman, and Josh Tenenbaum. 3d-aware scene manipulation via inverse graphics. *Advances in neural information processing systems*, 31, 2018.

[114] Minguk Kang, Woohyeon Joseph Shim, Minsu Cho, and Jaesik Park. Rebooting ACGAN: Auxiliary classifier GANs with stable training. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[115] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *International Conference on Learning Representations*, 2017.

[116] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017.

[117] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow. Adversarial autoencoders. In *International Conference on Learning Representations*, 2016.

[118] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016.

[119] Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. Mode regularized generative adversarial networks. *CoRR*, abs/1612.02136, 2016.

[120] Gao Huang, Yang Yuan, Qiantong Xu, Chuan Guo, Yu Sun, Felix Wu, and Kilian Weinberger. An empirical study on evaluation metrics of generative adversarial networks. 2018.

[121] Abhay Yadav, Sohil Shah, Zheng Xu, David Jacobs, and Tom Goldstein. Stabilizing adversarial nets with prediction methods. In *International Conference on Learning Representations*, 2018.

[122] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[123] L. Theis, A. van den Oord, and M. Bethge. A note on the evaluation of generative models. In *International Conference on Learning Representations*, Apr 2016.

[124] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[125] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. In Doina Precup and Yee Whye Teh,

editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2642–2651. PMLR, 06–11 Aug 2017.

[126] Partha Ghosh, Dominik Zietlow, Michael J. Black, Larry S. Davis, and Xiaochen Hu. Invgan: Invertible gans. *CoRR*, abs/2112.04598, 2021.

[127] Yizhe Zhang, Zhe Gan, and Lawrence Carin. Generating text via adversarial training. In *NIPS workshop on Adversarial Training*, volume 21, pages 21–32, 2016.

[128] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[129] Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. Adversarial learning for neural dialogue generation. In *EMNLP*, 2017.

[130] David Pfau and Oriol Vinyals. Connecting generative adversarial networks and actor-critic methods. *ArXiv*, abs/1610.01945, 2016.

[131] Diego Cabrera, Fernando Sancho, Jianyu Long, René-Vinicio Sánchez, Shaohui Zhang, Mariela Cerrada, and Chuan Li. Generative adversarial networks selection approach for extremely imbalanced fault diagnosis of reciprocating machinery. *IEEE Access*, 7:70643–70653, 2019.

[132] Xiang Li, Wei Zhang, Qian Ding, and Jian-Qiao Sun. Intelligent rotating machinery fault diagnosis based on deep learning using data augmentation. *Journal of Intelligent Manufacturing*, 31(2):433–452, 2020.

[133] Cheng Peng, Qing Chen, Longxin Zhang, Lanjun Wan, and Xinpan Yuan. Research on fault diagnosis of wind power generator blade based on sc-smote and knn. *Journal of Information Processing Systems*, 16(4):870–881, 2020.

[134] Siyu Shao, Pu Wang, and Ruqiang Yan. Generative adversarial networks for data augmentation in machine fault diagnosis. *Computers in Industry*, 106:85–93, 2019.

[135] Te Han, Chao Liu, Wenguang Yang, and Dongxiang Jiang. A novel adversarial learning framework in deep convolutional neural network for intelligent diagnosis of mechanical faults. *Knowledge-based systems*, 165:474–487, 2019.

[136] Magnus Wiese, Robert Knobloch, Ralf Korn, and Peter Kretschmer. Quant gans: deep generation of financial time series. *Quantitative Finance*, 20(9):1419–1440, 2020.

[137] Hung Ba. Improving detection of credit card fraudulent transactions using generative adversarial networks. *ArXiv*, abs/1907.03355, 2019.

[138] Eric Wu, Kevin Wu, and William Lotter. Synthesizing lesions using contextual gans improves breast cancer classification on mammograms, 2020.

[139] Stephan Schmidt, P Stephan Heyns, and Konstantinos C Gryllias. A discrepancy analysis methodology for rolling element bearing diagnostics under variable speed conditions. *Mechanical Systems and Signal Processing*, 116:40–61, 2019.

[140] Martha A Zaidan, Robert F Harrison, Andrew R Mills, and Peter J Fleming. Bayesian hierarchical models for aerospace gas turbine engine prognostics. *Expert Systems with Applications*, 42(1):539–553, 2015.

[141] Xiaohang Jin, Zijun Que, Yi Sun, Yuanjing Guo, and Wei Qiao. A data-driven approach for bearing fault prognostics. *IEEE Transactions on Industry Applications*, 55(4):3394–3401, 2019.

[142] Miao He and David He. Deep learning based approach for bearing fault diagnosis. *IEEE Transactions on Industry Applications*, 53(3):3057–3065, 2017.

[143] Haining Liu, Chengliang Liu, and Yixiang Huang. Adaptive feature extraction using sparse coding for machinery fault diagnosis. *Mechanical Systems and Signal Processing*, 25(2):558–574, 2011.

[144] Chunzhi Wu, Pengcheng Jiang, Chuang Ding, Fuzhou Feng, and Tang Chen. Intelligent fault diagnosis of rotating machinery based on one-dimensional convolutional neural network. *Computers in Industry*, 108:53–61, 2019.

[145] Yuyan Zhang, Xinyu Li, Liang Gao, Wen Chen, and Peigen Li. Intelligent fault diagnosis of rotating machinery using a new ensemble deep auto-encoder method. *Measurement*, 151:107232, 2020.

[146] Changqing Shen, Jiaqi Xie, Dong Wang, Xingxing Jiang, Juanjuan Shi, and Zhongkui Zhu. Improved hierarchical adaptive deep belief network for bearing fault diagnosis. *Applied Sciences*, 9(16):3374, 2019.

[147] Shengcai Deng, Zhiwei Cheng, Chuan Li, Xingyan Yao, Zhiqiang Chen, and René-Vinicio Sanchez. Rolling bearing fault diagnosis based on deep boltzmann machines. In *2016 Prognostics and System Health Management Conference (PHM-Chengdu)*, pages 1–6. IEEE, 2016.

[148] Chang Nho Cho, Ji Tae Hong, and Hong Ju Kim. Neural network based adaptive actuator fault detection algorithm for robot manipulators. *Journal of Intelligent & Robotic Systems*, 95(1):137–147, 2019.

[149] Huaqing Wang, Shi Li, Liuyang Song, and Lingli Cui. A novel convolutional neural network based fault recognition method via image fusion of multi-vibration-signals. *Computers in Industry*, 105:182–190, 2019.

[150] Qianli Ma, Enhuan Chen, Zhenxi Lin, Jiangyue Yan, Zhiwen Yu, and Wing WY Ng. Convolutional multitimescale echo state network. *IEEE Transactions on Cybernetics*, 2019.

[151] Guangzheng Hu, Huifang Li, Yuanqing Xia, and Lixuan Luo. A deep boltzmann machine and multi-grained scanning forest ensemble collaborative method and its application to industrial fault diagnosis. *Computers in Industry*, 100:287–296, 2018.

[152] Jinjiang Wang, Kebo Wang, Yangshen Wang, Zuguang Huang, and Ruijuan Xue. Deep boltzmann machine based condition prediction for smart manufacturing. *Journal of Ambient Intelligence and Humanized Computing*, 10(3):851–861, 2019.

[153] Kuei-Peng Lee, Bo-Huei Wu, and Shi-Lin Peng. Deep-learning-based fault detection and diagnosis of air-handling units. *Building and Environment*, 157:24–33, 2019.

[154] Haidong Shao, Hongkai Jiang, Xingqiu Li, and Tianchen Liang. Rolling bearing fault detection using continuous deep belief network with locally linear embedding. *Computers in Industry*, 96:27–39, 2018.

[155] Marsela Polic, Ivona Krajacic, Nathan Lepora, and Matko Orsag. Convolutional autoencoder for feature extraction in tactile sensing. *IEEE Robotics and Automation Letters*, 4(4):3671–3678, 2019.

[156] Gianluca D'Elia, Emiliano Mucchi, and Marco Cocconcelli. On the identification of the angular position of gears for the diagnostics of planetary gearboxes. *Mechanical Systems and Signal Processing*, 83:305–320, 2017.

[157] Martha A Zaidan, Andrew R Mills, Robert F Harrison, and Peter J Fleming. Gas turbine engine prognostics using bayesian hierarchical models: A variational approach. *Mechanical Systems and Signal Processing*, 70:120–140, 2016.

[158] Jaouher Ben Ali, Nader Fnaiech, Lotfi Saidi, Brigitte Chebel-Morello, and Farhat Fnaiech. Application of empirical mode decomposition and artificial neural network for automatic bearing fault diagnosis based on vibration signals. *Applied Acoustics*, 89:16–27, 2015.

[159] Ke Yan, Zhiwei Ji, Huijuan Lu, Jing Huang, Wen Shen, and Yu Xue. Fast and accurate classification of time series data using extended elm: Application in fault diagnosis of air handling units. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(7):1349–1356, 2017.

[160] Jamshed Iqbal, Raza Ul Islam, Syed Zain Abbas, Abdul Attayyab Khan, and Syed Ali Ajwad. Automating industrial tasks through mechatronic systems–a review of robotics in industrial perspective. *Tehnički vjesnik*, 23(3):917–924, 2016.

[161] F Caccavale, P Cilibrizzi, F Pierri, and L Villani. Actuators fault diagnosis for robot manipulators with uncertain model. *Control Engineering Practice*, 17(1):146–157, 2009.

[162] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[163] MY Gokhale, Daljeet Kaur Khanduja, et al. Time domain signal analysis using wavelet packet decomposition approach. *Int'l J. of Communications, Network and System Sciences*, 3(03):321, 2010.

[164] Lori Mann Bruce, Cliff H Koger, and Jiang Li. Dimensionality reduction of hyperspectral data using discrete wavelet transform feature extraction. *IEEE Transactions on geoscience and remote sensing*, 40(10):2331–2338, 2002.

[165] Chuan Li, René-Vinicio Sanchez, Grover Zurita, Mariela Cerrada, Diego Cabrera, and Rafael E Vásquez. Gearbox fault diagnosis based on deep random forest

fusion of acoustic and vibratory signals. *Mechanical Systems and Signal Processing*, 76:283–293, 2016.

[166] J Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[167] Fannia Pacheco, José Valente de Oliveira, René-Vinicio Sánchez, Mariela Cerrada, Diego Cabrera, Chuan Li, Grover Zurita, and Mariano Artés. A statistical comparison of neuroclassifiers and feature selection methods for gearbox fault diagnosis under realistic conditions. *Neurocomputing*, 194:192 – 206, 2016.

[168] Mark A Friedl and Carla E Brodley. Decision tree classification of land cover from remotely sensed data. *Remote sensing of environment*, 61(3):399–409, 1997.

[169] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[170] Pui Y Lee, Siu C Hui, and Alvis Cheuk M Fong. Neural networks for web content filtering. *IEEE intelligent systems*, 17(5):48–57, 2002.

[171] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[172] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

[173] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017.

[174] Wei Li, Xiang Zhong, Haidong Shao, Baoping Cai, and Xingkai Yang. Multi-mode data augmentation and fault diagnosis of rotating machinery using modified acgan designed with new framework. *Advanced Engineering Informatics*, 52:101552, 2022.

[175] Vasily Shapeev, Sergey Golushko, Vasily Belyaev, Luka Bryndin, and Pavel Kirillov. New versions of the least-squares collocation method for solving differential and integral equations. In *Journal of Physics: Conference Series*, volume 1715, page 012031. IOP Publishing, 2021.

[176] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. In *International Conference on Learning Representations*, 2018.

[177] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[178] Jae Hyun Lim and J. C. Ye. Geometric gan. *ArXiv*, abs/1705.02894, 2017.

[179] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *CoRR*, abs/1802.05957, 2018.

[180] Dustin Tran, Rajesh Ranganath, and David Blei. Hierarchical implicit models and likelihood-free variational inference. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[181] Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. Mode regularized generative adversarial. 2016.

[182] Alexander Levine and Soheil Feizi. Wasserstein smoothing: Certified robustness against wasserstein adversarial attacks. In *International Conference on Artificial Intelligence and Statistics*, pages 3938–3947. PMLR, 2020.

[183] Kimia Nadjahi. *Sliced-Wasserstein distance for large-scale machine learning: theory, methodology and extensions*. PhD thesis, Institut polytechnique de Paris, 2021.

[184] I. Deshpande, Z. Zhang, and A. Schwing. Generative modeling using the sliced wasserstein distance. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3483–3491, Los Alamitos, CA, USA, jun 2018. IEEE Computer Society.

[185] Ke Zhao, Hongkai Jiang, Chaoqiang Liu, Yanfeng Wang, and Ke Zhu. A new data generation approach with modified wasserstein auto-encoder for rotating machinery fault diagnosis with limited fault data. *Knowledge-Based Systems*, 238:107892, 2022.

[186] Dror Freirich, Tomer Michaeli, and Ron Meir. A theory of the distortion-perception tradeoff in wasserstein space. *Advances in Neural Information Processing Systems*, 34, 2021.

[187] Sajjad Piradl and Ali Shadrokh. Robust minimum distance estimation of a linear regression model with correlated errors in the presence of outliers. *Communications in Statistics-Theory and Methods*, 50(23):5488–5498, 2021.

[188] Ishan Deshpande, Ziyu Zhang, and Alexander G Schwing. Generative modeling using the sliced wasserstein distance. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3483–3491, 2018.

[189] Kimia Nadjahi, Alain Durmus, Pierre E Jacob, Roland Badeau, and Umut Simsekli. Fast approximation of the sliced-wasserstein distance using concentration of random projections. *Advances in Neural Information Processing Systems*, 34, 2021.

[190] Hao Tang, Hong Liu, Dan Xu, Philip HS Torr, and Nicu Sebe. Attentiongan: Unpaired image-to-image translation using attention-guided generative adversarial networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[191] Yongyi Lu, Yu-Wing Tai, and Chi-Keung Tang. Attribute-guided face generation using conditional cyclegan. In *Proceedings of the European conference on computer vision (ECCV)*, pages 282–297, 2018.

[192] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[193] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[194] Yongming Han, Shuang Liu, Di Cong, Zhiqiang Geng, Jinzhen Fan, Jingyang Gao, and Tingrui Pan. Resource optimization model using novel extreme learning machine with t-distributed stochastic neighbor embedding: Application to complex industrial processes. *Energy*, 225:120255, 2021.

[195] Tomoki Uemura, Janne J Näppi, Yasuji Ryu, Chinatsu Watari, Tohru Kamiya, and Hiroyuki Yoshida. A generative flow-based model for volumetric data augmentation in 3d deep learning for computed tomographic colonography. *International journal of computer assisted radiology and surgery*, 16(1):81–89, 2021.

[196] Liangwei Zhang, Jing Lin, Haidong Shao, Zhicong Zhang, Xiaohui Yan, and Jianyu Long. End-to-end unsupervised fault detection using a flow-based model. *Reliability Engineering & System Safety*, 215:107805, 2021.

[197] Haoliang Sun, Ronak Mehta, Hao H Zhou, Zhichun Huang, Sterling C Johnson, Vivek Prabhakaran, and Vikas Singh. Dual-glow: Conditional flow-based generative model for modality transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10611–10620, 2019.

[198] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.